40

Schriften aus der Fakultät Wirtschaftsinformatik und
Angewandte Informatik der Otto-Friedrich-Universität Bamberg

Reverse Engineering of Real-Time System Models from Event Trace Recordings

Andreas Sailer

University of Bamberg Press

40 Schriften aus der Fakultät Wirtschaftsinformatik und Angewandte Informatik der Otto-Friedrich-Universität Bamberg

Contributions of the Faculty Information Systems and Applied Computer Sciences of the Otto-Friedrich-University Bamberg

Schriften aus der Fakultät Wirtschaftsinformatik und Angewandte Informatik der Otto-Friedrich-Universität Bamberg

Contributions of the Faculty Information Systems and Applied Computer Sciences of the Otto-Friedrich-University Bamberg

Band 40

University of Bamberg Press
2019

Reverse Engineering of Real-Time System Models from Event Trace Recordings

Andreas Sailer

University
of Bamberg
Press
2019

Bibliographische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Informationen sind im Internet über http://dnb.d-nb.de/ abrufbar.

Diese Arbeit hat der Fakultät Wirtschaftsinformatik und Angewandte Informatik der Otto-Friedrich-Universität Bamberg als Dissertation vorgelegen.
1. Gutachter: Prof. Dr. Gerald Lüttgen
2. Gutachter: Prof. Dr. Jürgen Mottok
Tag der mündlichen Prüfung: 22.07.2019

Herstellung und Druck: docupoint, Magdeburg
Umschlaggestaltung: University of Bamberg Press
Umschlagbild: © Andreas Sailer

University of Bamberg Press, Bamberg 2019
http://www.uni-bamberg.de/ubp/

ISSN: 1867-7401
ISBN: 978-3-86309-690-8 (Druckausgabe)
eISBN: 978-3-86309-691-5 (Online-Ausgabe)
URN: urn:nbn:de:bvb:473-irb-465069
DOI: http://dx.doi.org/10.20378/irb-46506

Abstract

Model-driven approaches are experiencing an increasing acceptance in the automotive domain thanks to the availability of the AUTOSAR standard, which defines an open software architecture for the model-based development of real-time systems and a corresponding development methodology. However, the process of creating models of existing system components is often difficult and time consuming, especially when legacy code is involved or information about the exact timing is needed. The research community tackles this problem by developing algorithms for automatically deriving characteristics of the system's timing behaviour, e.g., response times and resource blockings from various artefacts such as source code or runtime measurements.

This work focuses on reversely engineering an AUTOSAR-compliant model, which can be used for further processing including timing simulation and optimisation, via a dynamic analysis from trace recordings of a real-time system. Although software reverse engineering via dynamic analysis has a long history, little research targets embedded systems and its use for multi-core architectures is largely unresearched. Furthermore, related work mainly discusses the analysis of individual characteristics of a real-time system, such as execution times or stimulation patterns instead of creating a description of the entire system. Huselius, whose work is among the publications most related to the topic of this thesis, proposes a technique to reverse engineer a model that reflects the general temporal behaviour of the original real-time software. However, like other existing solutions, it was not developed with AUTOSAR in mind. It is also not feasible to make this approach applicable to the automotive domain, because Huselius has not considered some required details, such as activation patterns, scheduling information, and compliance to the standardised development methodology of AUTOSAR.

We want to tackle this deficiency by introducing, in this work, an approach that seizes on Huselius's considerations and extends them in order to make them applicable to the automotive domain. To do so, we present CoreTAna, a prototypical tool that derives an AUTOSAR compliant model of a real-time system by conducting dynamic analysis using trace recordings. Its reverse engineering approach is designed in such a way that it fits seamlessly into the methodology specified by AUTOSAR. CoreTAna's current features are explained and their benefits for reverse engineering are highlighted, and a framework for evaluating the quality of synthesised models is

described. Motivated by the challenge of assessing the quality of reverse engineered models of real-time software, we also introduce a mathematical measure for comparing trace recordings from embedded real-time systems regarding their temporal behaviour and a benchmark framework based on this measure, for evaluating reverse engineering tools such as CoreTAna. This framework considers common system architectures and also includes randomly generated systems and systems of projects in the automotive domain and other industries. Finally, CoreTAna's performance and applicability are evaluated on the basis of this benchmark.

Kurzfassung

Dank der Verfügbarkeit des AUTOSAR Standards, welcher eine offene Softwarearchitektur zur modellbasierten Entwicklung von Echtzeitsystemen sowie eine korrespondierende Entwicklungsmethodik definiert, erfahren modellgetriebene Herangehensweisen eine steigende Akzeptanz in der Automobilbranche. Der Prozess zur Erstellung eines Models für bestehende Systemkomponenten ist jedoch oft beschwerlich und zeitaufwändig, vor allem wenn dabei bestehender Quellcode betroffen ist oder Information über den genauen zeitlichen Ablauf benötigt wird. Die Foschungsgemeinschaft geht dieses Problem durch die Entwicklung von Algorithmen an, die automatisch die Eigenschaften des Zeitverhaltens eines Systems, zum Beispiel Antwortzeiten und Blockierungen von Betriebsmittel, aus verschiedenen Artefakten, wie beispielsweise aus Quellcode oder Laufzeitmessungen, ableiten.

Diese Arbeit fokussiert sich auf die Rekonstruktion eines AUTOSAR-konformen Modells auf Basis einer dynamischen Analyse von Trace Aufzeichnungen aus einem Echtzeitsystem, welches zur Weiterverarbeitung in einer Echtzeitsimulation oder in einer Optimierung verwendet werden kann.

Obwohl die Nachbildung von Software mithilfe einer dynamischen Analyse auf eine lange Geschichte zurückschauen kann, zielen wenige Forschungsarbeiten auf eingebettete Systeme ab. Des Weiteren widmen sich verwandte Arbeiten hauptsächlich der Analyse von vereinzelten Eigenschaften eines Echtzeitsystems, wie zum Beispiel Ausführzeiten oder Aktivierungsmuster, anstelle eine Beschreibung des Gesamtsystems zu erstellen. Huselius, dessen Arbeit unter den Veröffentlichungen ist, die der Thematik dieser Abschlussarbeit am nächsten kommt, schlägt eine Technik vor die ein Modell rekonstruiert, das das generelle, zeitliche Verhalten der orginalen Echtzeitsoftware reflektiert. Wie andere bestehende Lösungsansätze wurde diese jedoch ohne Berücksichtigung des AUTOSAR Standards entwickelt. Es ist auch nicht möglich diesen Ansatz auf die Automobilbranche anzuwenden, da Huselius einige notwendige Details nicht berücksichtigt, wie zum Beispiel Aktivierungsmuster, Informationen über das Scheduling und die Konformität zur standardisierten Entwicklungsmethodik von AUTOSAR.

Wir möchten daher diese Defizite angehen indem wir in dieser Arbeit einen Ansatz vorstellen, der auf den Überlegungen von Huselius aufsetzt und diese erweitert, um sie in der Automobilbranche anwendbar zu machen. Hierfür stellen wir

CoreTAna vor, ein prototypisches Werkzeug, das ein AUTOSAR-konformes Modell eines Echtzeitsystems ableitet basierend auf der dynamischen Analyse von Trace Aufzeichnungen. Dessen Vorgehensweise bei der Rekonstruierung ist so entworfen, dass es sich nahtlos in die von AUTOSAR spezifizierte Methodik einpasst. CoreTAna's gegenwärtige Funktionalitäten werden erklärt und deren Vorteile für die Modellrekonstruktion hervorgehoben.Motiviert durch die Herausforderung die Qualität der rekonstruierten Modelle von Echtzeitsystemen zu bewerten, stellen wir ferner ein mathematisches Maß für den Vergleich von Trace Aufzeichnungen bezüglich ihrem zeitlichen Verhalten vor sowie eine Framework basierend auf diesem Maß zur Beurteilung von Werkzeugen wie CoreTAna. Dieses Framework berücksichtigt gebräuchliche Systemarchitekturen und auch zufällig generierte Systeme sowie Systeme aus Projekten in der Automobilbranche und anderen Industriezweigen. Schließlich werden CoreTAna's Leistungsfähigkeit und Anwendbarkeit anhand dieses Frameworks evaluiert.

Acknowledgements

This work was conducted in the context of the ITEA2 founded project AMALTHEA ("Model Based Open Source Development Environment for Automotive Multi Core Systems") and its successor AMALTHEA4public ("Enabling of Results from AMALTHEA and others for Transfer into Application and building a Community"). Both were pan-European projects that consisted of a mixture of academic and industrial partners whose goal it it was to develop a consistent, open, and expandable tool platform for engineering automotive multi-core systems.

Foremost, I would like to thank my industrial advisors Prof. Dr. Martin Hobelsberger and Dr. Michael Deubzer at Timing-Architects Embedded Systems GmbH for giving me the chance to work on this topic. We had a lot of inspiring discussions and they taught me proper research work and scientific writing. I am deeply grateful to Prof. Dr. Gerald Lüttgen for the great support and for giving me the opportunity to write this thesis in the Software Technologies Research Group at the University of Bamberg. I thank my further advisors Prof. Dr. Jürgen Mottok for educating and guiding me over the past years with great care, enthusiasm, and patience, and Prof. Dr. Ute Schmid for her help and the feedback she gave me over the years.

I thank my research colleague Stefan Schmidhuber for the inspiring discussions and the great time we had as Ph.D. students at the LaS^3. I am indebted to all colleagues at the LaS^3 and Timing-Architects, in particular Erna Oklapi, Erjola Lalo, Felix Martin, Jürgen Pöllinger, Maximilan Hempe, Ariane Maack, who have made my time very enjoyable and also helped me countless times.

So many friends have supported me over the last years and I want to apologise to any friend I may have omitted in that time. I am fortunate to have a loving and supportive family, who put great trust in me. They always believed in me and gave me a strong motivation.

January 2019, Regensburg

"You see there is only one constant. One universal.
It is the only real truth: Causality. Action, reaction. Cause and effect."

— The Matrix (1999)

Contents

1

Introduction

The AUTomotive Open System ARchitecture (AUTOSAR)[1] standard has experienced gradual acceptance in the automotive domain since its release of version 3.0 in 2007. One of its major goals is to define an open software architecture for the model-based development of real-time systems and a corresponding development methodology. Nevertheless, the process of deriving a model is often difficult, error-prone, and time consuming, especially when legacy code is involved [1]. The consideration of timing behaviour is an extra challenge that requires substantial effort [2], but is essential to the modelling process. Indeed, due to the current shift towards multi-core architectures in the automotive industry original equipment manufacturers (OEMs) are forced to gain detailed knowledge about their legacy software, including stimulation patterns and task execution times. Moreover, most tools which tackle the challenges implied by this shift to multi-core, such as finding an ideal task-to-core allocation or task priority assignment [3], require a model for analysis.

1.1. Motivation

Motivated by the challenges highlighted above the research community has considered a variety of approaches, including program slicing [4], machine learning [5], and purely mathematical approaches [6]. This resulted in the development of several solutions [7] over the past decade. Because each solution focuses on a specific characteristic of real-time systems different modelling approaches for the abstract description of, e.g., the system's architectural composition [8] or temporal [9, 10] and functional behaviour [8, 11] have emerged. However, there is no approach that has been developed with AUTOSAR in mind.

There are also many sources from which system information can be ac-

[1] `http://www.autosar.org`

quired [12]. Many existing approaches work on the basis of source code [4, 8, 10, 11]. However, this turns out to be infeasible in the automotive domain, because software functionality is typically provided by vendors which do not hand over source code. Furthermore, static analyses lack the ability to determine dynamic information such as execution times. Since the timing aspects of a system are of particular importance for the intended use cases presented later in Chapters 1.1.1 to 1.1.3, the most convenient way to automatically derive a model is by conducting a dynamic analysis based on traces. Trace recordings, or traces for short, store relevant events that were observed during a system's runtime.

Huselius [9] has conducted one of the latest and most extensive research concerning automatic modelling using execution-time recordings. Huselius proposes a technique to reverse engineer a model that reflects the general temporal behaviour of the original real-time software. However, like existing solutions, it was not developed with AUTOSAR in mind. It is also not feasible to make this approach applicable to the automotive domain, because Huselius has not considered some required details, such as activation patterns, scheduling information, and compliance to the standardised development methodology of AUTOSAR.

Thus, there is still a need for a solution that reverse engineers an AUTOSAR-compliant model from trace recordings of a real-time system. We want to tackle this deficiency by introducing, in this work, an approach that seizes on Huselius's considerations and extends them in order to make them applicable to the automotive domain. The main focus is on automatically synthesising the temporal behaviour of the software under investigation, in addition to an architectural system description, which can be done from scratch or by enriching an existing model. The latter is performed if system information is already available, e.g., from a static analysis of software artefacts such as source or object code.

The fact that our solution creates an AUTOSAR-compliant artefact, which can be used for further processing, improves the interoperability considerably for existing and established work environments. Thus, this not only enables a new way to integrate legacy software into the automotive development process, but it also makes further use cases applicable including the following ones.

1.1.1. Documentation

Creating a model in an automatic manner actively supports the documentation of a software project during the entire development process. For example, in a model-based development process, manual changes to the software no longer have to be applied to the model in an often error-prone and time-consuming manner. Indeed, some pieces of information cannot be handled manually at all or only very costly due to their complexity, such as the stimulation patterns of interrupts. Keeping a model up-to-date is also much more expedient, e.g., when *studying* the model to understand the system. However, also in cases in which the documentation is used for *information exchange* between an OEM and its Tier-1s, a model containing the latest information is invaluable. Finally, a model can be used for the *validation* of the underlying system. Since the model reflects the system's actual behaviour, this information, e.g., the response times, can be compared to the desired behaviour specified by the user requirements.

1.1.2. Simulation

The reverse engineered model can be used for timing simulation, in order to make a statement about the dynamic software behaviour of the real-time system under investigation. Similar to *debugging*, it is possible to analyse a system in detail and to identify situations leading to incorrect behaviour and their causes. A simulation can also be used to *test* whether the system responds correctly to a particular input.

Another application is the assessment of a system's *performance* by simulating its schedulability, so that characteristics of the system's behaviour, e.g., response times and resource blockings can be examined and considered for further system optimisation. Therefore, timing simulation also allows one to perform an *impact analysis*, where the behavioural impact of system changes and its resulting performance is checked. This helps engineers to not only examine and understand the system in full detail, but also to manually *refine* the system step-by-step to optimise its performance.

1.1.3. Optimisation

A model can also help the automatic optimisation of a system. To do so, the aims of the optimisation have to be defined in terms of real-time metrics such as response

times or utilisations. The characteristics of the system are then altered by modifying the model, and an evaluation of the system's performance using, e.g., a timing simulation enables engineers to assess whether the modifications have a positive or negative impact on the system's behaviour, especially regarding the targeted real-time metrics.

1.2. Contributions

Motivated by the lack of a solution that reverse engineers an AUTOSAR-compliant model from the dynamic analysis of a real-time system, the following research questions are outlined:

Question Q1:

To what extent is it possible to automatically synthesise an AUTOSAR-compliant model of a real-time system that covers the system's temporal behaviour based on event trace recordings?

Question Q2:

Can a synthesised model be validated with regards to the extent in which its representation reflects the temporal behaviour of the corresponding actual system?

Question Q3:

To what extent is an approach for the automatic model synthesis of a real-time system from event trace recordings applicable to industrial projects in the automotive domain?

Although the aforementioned research questions can be partly solved with existing solutions, our approach creates added value. This work makes in detail the following contributions, which altogether cover the aforementioned research questions:

Contribution C1:

A set of algorithms that automatically synthesise an AUTOSAR-compliant model with details on the system's timing behaviour based on event trace recordings. (Realisation of Q1)

The approach presented by Huselius [9] provides some algorithms that are also relevant for synthesising an AUTOSAR-compliant model. For this reason, the solution approach described in this work takes Huselius's considerations and extends them to make them applicable to the automotive domain. For example, the assumptions that are made by Huselius such as knowing the employed scheduling algorithm are

eliminated and additional algorithms including those for comprehending a system's dynamic stimulation are introduced.

Contribution C2:

An approach that assesses the extent in which a model reflects the temporal behaviour of its corresponding actual system. (Realisation of Q2)

An approach is introduced that allows one to make a statement about the reverse engineered model. By employing a model-based timing simulation, it is verified whether the simulation traces of the reverse engineered model and the hardware traces of the system under investigation show the same temporal behaviour.

Because the quality of reverse engineering and, thus, of the resulting model heavily depends on the provided input, trace recordings that monitor the system at different levels of detail are used. This not only highlights the sensitivity of the developed algorithms but also allows one to estimate prior to reverse engineering the quality of the resulting model based on the characteristics of a given trace such as its length.

Contribution C3:

A metric expressing the accordance of two sample event trace recordings with respect to their represented temporal behaviour. (Realisation of Q2)

We define a new measure called *Distance of Timed Actions (DoTA)* which assesses the quality of reverse engineered real-time software based on the *Euclidean Distance*. It determines how well the resulting model reflects the actual system by comparing real-time metrics, e.g., activate-to-activate times and response times of tasks that were determined from the simulation traces of the reverse engineered model and the hardware traces of the system under investigation. Further use cases for this measure show that it is beneficial in other situations.

Contribution C4:

An extensible realisation of the developed algorithms to automatically synthesise a probabilistic system model from event trace recordings that fits fluently in the AUTOSAR development process and that supports the industrial use cases. (Realisation of Q3)

One of the main outcomes of this work is *Core Trace Analyser (CoreTAna)*, a novel tool that reverse engineers an AUTOSAR-compliant model of a real-time, single- or multi-core system, including its exact timing behaviour, based on the dynamic analysis of the system's trace recordings. CoreTAna can derive such a model either automatically

from scratch or by enriching an existing model. It is part of the TA Tool Suite as an experimental module and is currently in transition to becoming a mature product. During the work on this thesis, CoreTAna has already been employed by co-workers and customers in many industrial projects. This has allowed us to not only integrate it flawlessly with the AUTOSAR development process, but also to identify the features that are in great demand such as analysing a system's dynamic stimulation.

Contribution C5:

> *Case studies that on the one hand, show the correctness of the developed algorithms and on the other hand, the applicability and usefulness of the implementation for actual industrial projects.* (Realisation of Q3)

Based on Huselius's idea of varying common architectural patterns in the real-time software domain to show the capability of the developed algorithms, a synthetic benchmark is presented. This benchmark extends existing work by considering AUTOSAR-specific aspects such as Exclusive Areas and covers the functionality provided by CoreTAna.

In addition to this benchmark, four industrial case studies are conducted. Three of these illustrate the use of CoreTAna in the automotive domain. Thereby, the reverse engineering copes in each case study with a trace recording that stores information about the system at a specific level of detail. Finally, CoreTAna is also applied to a project in the telecommunication domain to show that the developed algorithms are not only limited to automotive domain.

1.3. Research Method

In this section the research method is presented, which was applied to the research questions defined above. It sets itself apart from others because the research was performed within publicly funded projects, *AMALTHEA* and its successor *AMALTHEA4public*[2], and in collaboration with an industrial company, *Timing-Architects Embedded Systems*[3]. The research described in this work arose from actual industrial problems and has been continuously influenced by the feedback and experience given by both project partners and customers. An overview of this synergy and the embedded research method is depicted in Fig. 1.1.

[2] `http://www.amalthea-project.org`
[3] `http://www.timing-architects.com`

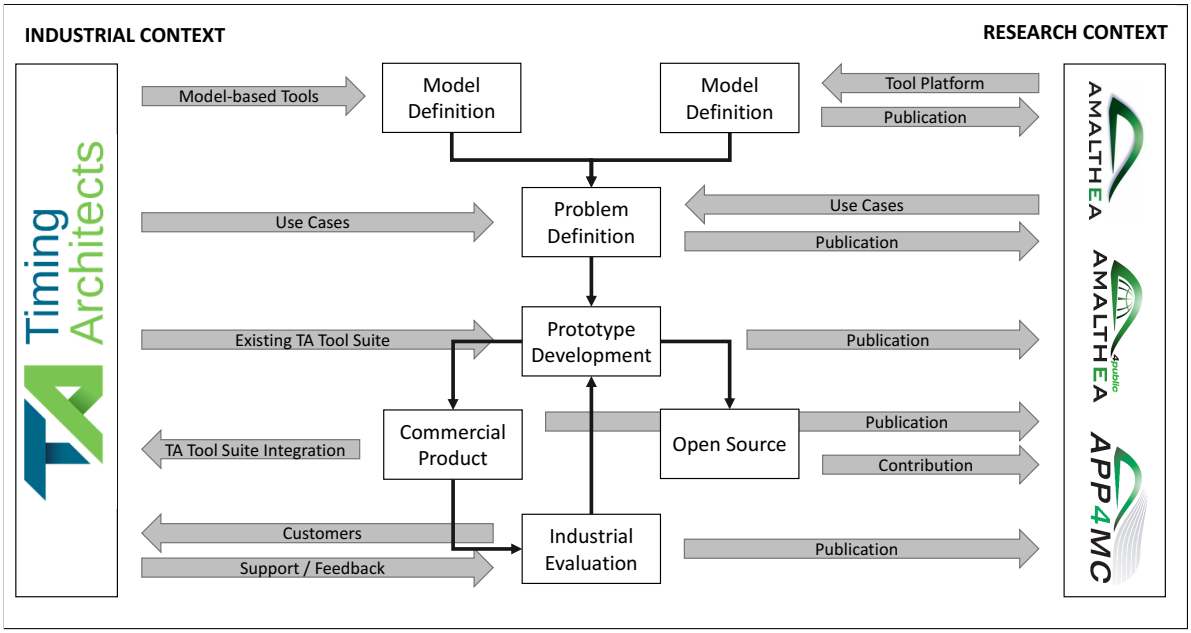


Figure 1.1.: Research Method. Overview of the Applied Research Method.

The research problems introduced in Chapter 1.1 were originally identified by Timing-Architects (TA), a company which offers model-based tools for the system design, simulation, automated optimisation, and target verification of embedded real-time multi-core and many-core systems. When the company was founded in 2011, model-based development had not received general acceptance in the automotive domain. So, already in the early years, it became obvious that obtaining a model of the customer's software system based on which TA's tools work, is one of the crucial points in the success of the start-up. Near the same time the pan-European research project AMALTHEA started working on a customised, open source tool chain platform for the development of embedded multi-core systems, including an AUTOSAR-compliant model for data exchange. This working environment influenced the problem definition in such a way that the meta-model of the models which have to be re-engineered were given as well as the technical approach, namely the use of a dynamic analysis. The latter is set because TA, as a tool vendor and consulting company, does not get access to their customer's source code.

Although the models targeted for reverse engineering are applicable to every embedded domain, the developed prototype focuses on solving research problems that are especially relevant to the automotive industry. This is because the developed algorithms tackle the use cases provided by customers of TA or by partners of the AMALTHEA project, which are mainly from the automotive domain. Nevertheless,

the algorithms and the developed prototype are valid for the entire embedded domain. However, applying them to another domain may require one to look into additional system characteristics that do not apply to the automotive domain, such as a different scheduling algorithm.

The algorithms developed for the prototype, which served as a proof of concept, found its way into two different implementations. One version was turned into a commercial product by integrating it into the existing TA Tool Suite. Another version is included in the AMALTHEA open source platform, in which it serves as a link between trace recordings and the corresponding system model to achieve a continuous development tool chain. In contrast to the commercial version, the open source contribution does not contain all the developed algorithms. Because the open source version serves as an example implementation, it solely analyses trace recordings at the process level. In the meantime, the open source platform AMALTHEA transitioned to the Eclipse Foundation and is now an official Eclipse project called *Application Platform Project for Multi-Core (APP4MC)*. This ensures that the developed algorithms continue to be maintained and gives a chance for other embedded domains to get feedback.

Finally, the industrial connections represented by customers of TA on the one side and partners from the AMALTHEA project on the other side were not only consulted regarding use cases but also for thorough evaluations. Thus, it was possible to not only test the developed algorithms, but also the implementation within actual industrial environments and to get detailed feedback, which again influenced further developments.

The conducted research and the implemented prototype gained benefit not only for TA but also for the research project in the form of disseminations. Over the years the following publications, motivated by the scope of this work, were released:

Main Publications

- **A. Sailer, M. Deubzer, G. Lüttgen and J. Mottok, "Comparing Trace Recordings of Automotive Real-time Software", in *Intl. Conf. on Real-Time Networks and Systems*, ACM, 2017**

  Motivated by the challenge of assessing the quality of reverse engineered models of real-time software, we present a novel mathematical measure for comparing trace recordings from embedded real-time systems regarding their temporal behaviour. We also introduce a benchmark framework based on this

measure, for evaluating reverse engineering tools such as CoreTAna. This considers common system architectures and also includes randomly generated systems and three systems from industrial automotive projects. Finally, an industrial case study demonstrates other use cases of our measure, such as impact analysis.

- **A. Sailer, S. Schmidhuber, M. Hempe, M. Deubzer and J. Mottok, "Distributed Multi-Core Development in the Automotive Domain – A Practical Comparison of ASAM MDX vs. AUTOSAR vs. AMALTHEA", in *First Multi-Core Safe and Software-intensive Systems Improvement Community Workshop*, VDE, 2016**
  To give a state-of-the-art overview of distributed multi-core development in the automotive industry, we present a comparison of three standards that are commonly used in this domain: ASAM MDX, AUTOSAR, and AMALTHEA. To do so, we oppose the defined system models of each standard, their methodology, and reference implementation with each other. A case study on consolidating software functions between car manufacturers and their suppliers shows the application of each standard within an actual industrial multi-core project and highlights their strengths and limitations.
- **A. Sailer, M. Deubzer, G. Lüttgen and J. Mottok, "CoreTAna: A Trace Analyzer for Reverse Engineering Real-Time Software", in *Intl. Conf. on Software Analysis, Evolution, and Reengineering*, IEEE, 2016, pp. 17–28**
  We present CoreTAna, a novel tool that reverse engineers an AUTOSAR-compliant model of a real-time system from a dynamic analysis of its trace recordings. This paper gives an overview of CoreTAna's current features and discusses its benefits for reverse engineering.
- **A. Sailer, "Towards an Automated Reverse Engineering of Design Models from Trace Recordings", in *Jahrestagung der Gesellschaft für Informatik*, GI, 2014, pp. 2233–2245**
  This research aims to extract, analyse, and deduce information about the system under observation from a limited view of the internal details of the system. To achieve this, an iterative approach consisting of three steps is proposed.
- **A. Sailer, S. Schmidhuber, M. Deubzer, M. Alfranseder, M. Mucha and J. Mottok, "Optimizing the Task Allocation Step for Multi-Core Processors within AUTOSAR", in *Intl. Conf. on Applied Electronics*, IEEE, 2013, pp. 247–252**
  This paper focuses on a model-based optimization approach for the task allocation problem in embedded multi-core systems. The starting point is the sys-

tem description in AUTOSAR and runtime measurements of the runnables in hardware traces. Based on this, an initial software partitioning of runnables to tasks is created. Subsequently, we use a genetic algorithm to create and evaluate solutions to the task allocation problem.

- **A. Sailer, S. Schmidhuber, M. Deubzer and J. Mottok, "AMALTHEA – Plattform für kontinuierliche, modellbasierte Entwicklung", in *Embedded Software Engineering Kongress*, ELEKTRONIKPRAXIS Vogel Business Media GmbH & Co. KG und MicroConsult Microelectronics Consulting & Training GmbH, 2013, pp. 538–544**
  We give an overview of AMALTHEA, an open source tool platform for engineering embedded multi- and many-core software systems. The Eclipse-based platform enables the creation and management of complex tool chains including simulation and supports interoperability via a unified, AUTOSAR-compliant model.

Supervised Theses

- **P. Harrer, "Development of an Algorithm for Comparing Traces", Master's Thesis, Hochschule Nordhausen, 2016**
  This master thesis describes the conceptual design and the development of different algorithms for comparing hardware traces. It introduces real-time metrics based on which the algorithms evaluate the similarity of traces.
- **F. Martin, "Transformation of Hardware Traces to System Traces for Embedded Multi-Core Real-Time Systems", Master's Thesis, Ostbayerische Technische Hochschule (OTH) Regensburg, 2015**
  This thesis examines different trace techniques and analyses the coherence between hardware, software, and system level entities. Based on the results, a mapping from software level to system level is introduced and validated via hardware tracing.
- **X. Tang, "Trace-based Timing Verification of Real-Time Systems", Master's Thesis, University of Shanghai for Science and Technology, 2014**
  This thesis introduces a round-trip approach for real-time operating systems, i.e., from model to source code and back again via execution and hardware tracing. Its motivation is to verify the results from a timing simulation by comparing them with the timing behaviour of the actual hardware. A tool was implemented to highlight differences between two models regarding their timing behaviour.

Peer-reviewed Publications

- **M. Alfranseder, M. Mucha, S. Schmidhuber, A. Sailer, M. Niemetz and J. Mottok, "A Modified Synchronization Model for Dead-lock Free Concurrent Execution of Strongly Interacting Task Sets in Embedded Systems", in *Intl. Conf. on Applied Electronics*, IEEE, 2013, pp. 13–18**
  We apply global scheduling algorithms to embedded, real-time, multi-core systems. To achieve this, a new resource model and locking mechanism are introduced. The locking mechanism is then compared to existing ones regarding the timing effects of their blocking behaviour.
- **J. Mottok, M. Alfranseder, S. Schmidhuber, M. Mucha and A. Sailer, "How to Improve the Reactiveness and Efficiency of Embedded Multi-core Systems by Use of Probabilistic Simulation and Optimization Techniques", in *NATO Advanced Research Workshop on Improving Disaster Resilience and Mitigation – IT Means and Tools*, Springer, 2013, ch. 16, pp. 253–268**
  This book chapter addresses different challenges of embedded multi-core real-time systems. On the one hand, we discuss a deadlock-free synchronization model. On the other hand, we present a model-based approach to map the tasks of an embedded real-time system to the cores of a multi-core processor, and to derive an execution time model from runtime measurements of software functions. Based on this information, an optimisation technique improves the system's task-to-core mapping by performing probabilistic simulations.

1.4. Outline

The remainder of this thesis is structured into three parts as follows. Part I introduces the necessary background information for this work. It begins with an overview of the related work on reverse engineering. Chapter 2 covers the latest research in the reverse engineering on embedded software and those in other software domains including web services. Finally, similar research questions from domains which are not related to software, such as work flow and process mining, are discussed. Chapter 4 continues and focuses on the meta-model that is used for the results of the reverse engineering. It introduces *AMALTHEA*, the AUTOSAR-compliant model generated by our reverse engineering tool, and details how specific characteristics of a real-time systems, including hardware and software, are modelled. Next, an extensive overview on

Table 1.1.: Research Questions. Overview of the research questions and thesis contributions addressed by each chapter.

Chapter	Question			Contribution				
	Q1	Q2	Q3	C1	C2	C3	C4	C5
1								
2	✓							
3								
4								
5								
6	✓		✓	✓			✓	
7		✓			✓	✓		
8			✓				✓	✓
9								

trace recording is given in Chapter 5. This chapter begins with the state-of-the-art in trace recording by presenting different trace techniques and how they can be categorised. It continues with an introduction to *Best Trace Format (BTF)*, the trace format which is processed by our reverse engineering algorithms. The individual events that are defined by this trace format for recording the temporal behaviour of a real-time system during runtime are described in detail.

Part II elaborates on the technical contributions of this thesis. Table 1.1 gives an overview of the research questions and thesis contributions addressed by each chapter. Chapter 6 focuses on CoreTAna, our reverse engineering tool. CoreTAna allows one to automatically synthesise a probabilistic system model from event trace recordings. This chapter elaborates on CoreTAna's internal workings by detailing its approach and introducing its algorithms. The algorithms are discussed separately, which allows one to get an exact overview on the events that are required as input for each algorithm and, thus, for reversely engineering individual aspects of the system model. Chapter 7 lays the foundation for evaluating the algorithms. The requirements for determining how well a synthesised model reflects the timing behaviour of the original system are elicited. This is done by introducing a synthetic benchmark, which defines common architectural patterns in the real-time software domain and feasible variations for each pattern. Based on this benchmark, CoreTAna's perform-

ance is evaluated. After the latest research regarding the comparison of trace recordings is presented, Chapter 7 continues with the definition of a novel measure. This measure, called DoTA, is validated with the synthetic benchmark and further uses cases are highlighted. Finally, Chapter 8 addresses the evaluation of the developed algorithms and CoreTAna's applicability in industrial projects. It consists of three different parts. First, CoreTAna's performance is determined based on the aforementioned benchmark. Second, an evaluation with randomly generated systems is performed. Third, case studies document our experiences with CoreTAna's use in actual industrial projects. Each case study considers a typical system in the automotive domain, which are kindly provided by industrial partners of the AMALTHEA project or by customers of TA.

In Part III, a summary of the thesis achievements are given and future research directions are highlighted.

Part I.

Background

2

Related Work

Reverse engineering via dynamic analysis has a long history. For example, Biermann researched the inference of Turing machines from sample computations already in 1972 [23]. Cornelissen et al. give in [24] an extensive overview on research literature which concerns program comprehension via dynamic analysis. Different facets including activity, target, method, and evaluation are used to characterise the articles of interest. Amongst the examined articles the analysis of a system's behaviour, the understanding of execution traces, and the recovery of high-level designs have been found to be the least performed activities. The target facet reflects the intended field of application. Cornelissen et al. conclude that "dynamic analysis is rarely applied to legacy software systems" [24, p. 11] and that the use of dynamic analysis on multi-core architectures is "largely unexplored territory" [24, p. 11]. Finally, the article's evaluation outlines the manners in which the conducted research was validated. Here, the authors "found industrial evaluations to be uncommon" [24, p. 12].

In addition, Kienle et al. provide in [7] a survey on reverse engineering, but focus on its use in embedded systems. The authors review various reverse engineering techniques and tools that are applicable for the specific characteristics of real-time systems and cluster them into categories according to the resulting artefacts, such as state machines, architecture models, and simulation models. They conclude that "little research in reverse engineering targets embedded systems" [7, p. 8].

In the following, a selection of existing reverse engineering techniques is presented in more detail. At first, publications most related to the topic of this thesis are discussed by laying out techniques that are applicable in the domain of embedded software. This is followed by more general techniques and approaches from other domains. Table 2.1 overviews what is covered by each related work that is presented in this section.

Table 2.1.: Related Work. Overview on what is covered by each related work that is discussed in this section

		Scheduling Properties	Stimulation Patterns	Runtime Behaviour	Static Analysis	Dynamic Analysis	Hybrid Analysis	Source Code	Byte Code	Trace Recording	Approach	Algorithm	Benchmark	Case Study
		Model Synthesis			Analysis Type			Input			Contribution			
This thesis		**X**	**X**	**X**		**X**			**X**	**X**	**X**	**X**	**X**	
Huselius	[25]		X	X		X		X	X	X				X
	[26]			X		X		X	X	X	X			
	[27]			X		X		X	X	X	X			X
	[9]		X	X		X		X	X	X	X	X	X	X
Kraft	[28]		X	X			X	X		X	X			
	[12]			X			X	X		X	X	X		
	[4]			X			X	X		X	X	X		X
Sirofakis	[29]			X	X			X			X			X
Thiele	[6]		X	X		X				X		X		
	[30]		X	X		X				X		X		
	[31]		X	X		X				X	X	X		X

		Scheduling Properties	Stimulation Patterns	Runtime Behaviour	Static Analysis	Dynamic Analysis	Hybrid Analysis	Source Code	Byte Code	Trace Recording	Approach	Algorithm	Benchmark	Case Study
		Model Synthesis			Analysis Type			Input			Contribution			
Altenbernd	[10]			X	X		X	X		X		X		
Ciccozzi	[32]			X	X		X	X	X	X				X
	[33]			X	X		X	X	X	X				X
	[11]			X	X		X	X	X	X	X			X
Terrasa	[34]			X	X		X	X	X	X				X
Auguston	[35]			X	X		X	X		X				
Murphy	[36]			X	X		X	X		X				X
	[37]			X	X		X	X		X	X			
Van Hoorn	[38]			X	X		X	X	X	X				
Krogmann	[5]			X	X			X	X	X	X	X		
	[8]			X		X	X	X	X	X	X	X	X	X
Cook	[39]			X	X				X	X	X	X		
	[40]			X	X				X	X	X	X		

	Scheduling Properties	Stimulation Patterns	Runtime Behaviour	Static Analysis	Dynamic Analysis	Hybrid Analysis	Source Code	Byte Code	Trace Recording	Approach	Algorithm	Benchmark	Case Study
	Model Synthesis			Analysis Type			Input			Contribution			
van der Aalst [41]			X	X			X	X	X				
Wen [42]			X	X			X	X	X	X			

2.1. Embedded Software Domain

In general, the embedded systems domain is equivalent to that of real-time systems. This means that for embedded systems the correctness of its behaviour depends not only on functional correctness but also on temporal accuracy. This fact imposes extra challenges on the process of reverse engineering which is why generic reverse engineering techniques cannot be applied without further ado. For that reason, this section presents the related work in reverse engineering techniques that are applicable for the specific characteristics of real-time systems.

Huselius's contributions. Contributions with the closest relation to the research problem discussed in this thesis are provided by Huselius [9, 25–27]. In [25], a method for automatic model generation based on observations from a running system is introduced; additional technical details on this approach can be found in [26]. In [27], the presented approach is extended regarding the aspects of model validation. Finally, a complete overview on the research topic of dynamic model extraction is given in [9].

The approach proposed by Huselius uses instrumentation to record events during a system's runtime. The following events are monitored: "send and receive operations providing interprocess communications (ipc), variable updates, and context switches" [27, p. 2]. Based on these events, the actions performed by the system are detected, such as send and receive operations of inter-process communication, end of a job, value updates to variables, and execute statements. However, instrumentation cannot only alter the system's timing but can also change functional behaviour [43]. To prevent this so-called probe effect, the author suggests to reduce the overhead of recording. For that reason, not all variable updates are recorded but only "those that represent the data-state in the system" [9, p. 54]. This results in the problem, which is discussed in Huselius's case study, that often several attempts are necessary to find a sufficient probing. Another drawback of the presented approach is that instrumentation requires access to the implementation code and the possibility to modify it.

Within the scope of this thesis, working on the basis of source code is infeasible, because software functionality is often provided by vendors that do not hand over source code. For this reason and also to avoid the aforementioned probe effect, hardware tracing is used. A high-speed trace interface and the availability of dedicated co-processors and memory for tracing on the CPU allows engineers to record more pieces of information on the system's runtime behaviour without any temporal im-

pact on its execution. That way, also a larger range of events can be considered when compared to Huselius's approach.

The approach by Huselius synthesises a model that can describe the observed behaviour, namely "a probabilistic state-machine model expressed in the ART-ML language" [25, p. 1]. ART-ML is an architecture and real-time behaviour modelling language for describing the temporal behaviour of real-time software systems including their timing requirements and intended for analysis within a simulation environment. A complete definition of ART-ML can be found in [28, p. 125 ff.]. An ART-ML model consists of a set of tasks. Each task is defined by a set of attributes and a behavioural description. The former includes a specification of the tasks scheduling priority and its activation offset and recurrence. The latter describes the temporal and functional behaviour of the corresponding task in an abstract manner. For this purpose, expressions such as message box communication, binary semaphore access, or execution time consumption as well as the control flow statements of the programming language C and a probabilistic selection statement are available.

In contrast to the model used in this article, in which the behaviour of functions and data accesses can also be modelled, the "ART-ML model provides a very high-level view of the system" [9, p. 57] This additional level of detail is necessary to enable more use cases for the synthesised model, such as optimising the call sequence of functions within tasks.

Kraft's contributions. Kraft, formerly Andersson and a colleague of Huselius at that time, researched on how dynamic analysis can be used to model the temporal behaviour of embedded systems [4, 12, 28]. The authors propose different ways to extract the information necessary to facilitate the understanding and modelling of real-time software systems [28], but do not provide algorithms for automatically generating a model. With this basic research and the specification of ART-ML [28, p. 125 ff.], Kraft prepared the ground for the aforementioned approach proposed by Huselius and presented a strategy for extracting a simulation model from a real-time software system in [12]. This strategy combines Huselius's model synthesis approach with a hybrid model extraction method. The former is used on trace recordings of a system to automatically generate models for all tasks that are observed in the recordings. The resulting models are then inspected regarding their complexity. For complex tasks, the proposed hybrid model extraction is applied. At first, the source code of the system under investigation is used to extract static information for the model, such as

call-graphs and functional statements. For this purpose, a program slicing method called 'Katana' is used, which is presented in [43, p. 103 ff.]. The resulting model is then refined by dynamic aspects, including execution times and estimated probabilities for conditions, to obtain an accurate model for simulation.

Because Kraft's method represents a hybrid approach, the availability of source code is required. If we consider just the dynamic analysis, the informative content added by this is merely the extraction of the execution time and the estimation of probabilities. Thus, the added value by this work turns out to be negligible for our research.

Sirofakis's contributions. Another problem which is similar to the topic of this thesis is discussed in [29]. Sirofakis et al. describe a modelling framework for building timed models of synchronous real-time systems. To remove divergences between an application software and its implementation, a relation between both is established by adding timing constraints to the application software. At first, the application software, which is represented by an Esterel program, is annotated with intervals defining the best-case and the worst-case execution time (BCET, resp., WCET). These times are estimated using existing techniques, such as profiling or static analysis. The program together with a model of its environment and a corresponding event sequence to control the environment is then compiled to C code. By doing so, the actual execution times of functions are substituted by the intervals provided by the annotations. The instrumented C code represents a timed automaton model of the system, which can be analysed via timing analysis techniques for relevant real-time properties. The problem discussed in [29] diverges from that targeted in this thesis that [29] only considers synchronous real-time systems. Furthermore, the proposed method is limited to single-processor implementations [29, p. 3] and, once again, annotations to the program have to be possible.

Thiele's contributions. In contrast to the literature discussed so far, the following works do not really focus on reverse engineering. Instead, the research group of Thiele define a mathematical framework, the so called *real-time calculus*, for analysing system properties of embedded systems [6, 30, 31].

Two models form the basis of the underlying mathematical framework and consequently of the system's analysis. On the one hand, this is the *event model*, which describes the minimum and maximum numbers of events that arrive within any time

interval. For that reason, the resulting functions are called the upper and lower arrival curve. The second model captures the processing capabilities of the system and works in a similar manner. In the so called *service curve*, the lower and upper processing limits of a resource are defined. Thus, the system is described on the basis of how much load is incoming and how much load can be processed. Although there is a method available to generate event traces for simulation or physical measurements [44], the models are just suitable for quantitatively characterising a system. For this reason, we also use the concept of arrival curves in CoreTAna to describe stimulation patterns of tasks. Other characteristics of the system such as the task priorities, however, require a qualitative specification.

Altenbernd's contributions. More recently, Altenbernd et al. [10] have developed an approach to automatically derive source-level timing models. Their goal is to infer a model that can be used for static worst case execution time (WCET) analysis or simulation in order to estimate the timing behaviour of a system at source-level. The resulting linear timing model specifies a correlation between execution times and virtual instructions, which represent an abstraction of the actual source code. For each virtual instruction the execution times are automatically identified from the measured execution times and recorded instruction counts. This is done by compiling example programs and executing these on the target hardware or a simulator. Thereby, the execution time needed to execute an example program is determined. In a next step, each example program is translated into the virtual instruction code format. A WCET analysis tool which establishes a relation between virtual instructions and their execution times, determines then the execution counts for all virtual instructions in an example program. Finally, the system under investigation is translated into the virtual instruction code format and, based on this together with the previously determined knowledge on virtual instructions and their execution times, estimations on the WCET and the execution time for a single program run can be calculated.

This approach differs from the one presented in this thesis by deriving source-level timing models. Not only does this approach require the availability of source code, it also reveals the internal program logic of the system under investigation by using a source-level timing model. Both of these drawbacks, however, are infeasible within the scope of our work.

Ciccozzi's contributions. Another related work is the round-trip approach introduced by Ciccozzi et al. [11, 32, 33], where target code is generated from design models according to the concept of model-based engineering. This step is often considered as a one-way street, where changes in the target code are not propagated back to the modelo. Furthermore, in embedded systems, characteristics such as resource consumption or execution time play a crucial role in the correct behaviour of the system. However, values for these characteristics are basically non-existent at the modelling level until the design is implemented and run on a specific platform. For this reason, Ciccozzi et al. introduce a round-trip approach which includes the automatic annotation of design models via code execution monitoring that is extended by aspects of multi-processing in [11, 33].

Instead of just generating source code from design models, the proposed approach inserts additional traceability information to establish a mapping between model elements and generated code segments. This allows one to automatically monitor and measure properties of a system during runtime including execution time, response time, heap and stack memory usage and then propagate the results back to the design model for further evaluation. For the definition of the design model, the CHESS modelling language [11, p. 5 ff.], an UML profile including tailored subsets of SysML and MARTE, is employed by Ciccozzi et al.. The result of the monitoring is a four-column log file that contains the aggregated value for each property and model element.

In general, the approach by Ciccozzi et al. is in line with the goal of our work, namely to automatically infer a model of a system which accurately reflects the system's runtime behaviour. But their approach already starts with an existing design model that describes the complete system. However, a developer has to handle multiple diverse models, of which each one covers a distinctive part of the system such as functional behaviour or architectural description. For this reason, the generation of an initial model, which can then be gradually enhanced, plays a crucial role and has not been discussed by Ciccozzi et al..

Terrasa's contributions. Motivated by the fact that statically analysing a real-time system, e.g., determining the WCET yields highly pessimistic results, Terrasa et al. present in [34] a framework for extracting temporal properties from real-time systems by analysing run-time traces. The temporal properties considered are either system related, e.g., the utilisation or task related with properties such as response times, computation times, blocking times, and jitter factors. The framework which

they introduce consists of two consecutive steps. At first, each temporal property is defined as a function over pre-defined sequences of state transitions. Then, based on a trace recording that can origin from hardware or software tracing the system evolution is reconstructed as state transitions. That way the temporal properties are determined. Finally, the authors present a case study in which their framework is used to extract temporal properties from a POSIX real-time application running on RT-Linux.

Although the deterministic finite automata, based on which the temporal properties are defined, are similar to those used in this work and could be adapted accordingly, the framework calculates just a set of performance metrics. A model of the overall system and the interaction of its elements, which can be considered for further system optimisation and which is required for our work, is not available.

2.2. Other Software Domains

While the previous section focuses on existing solutions for the embedded software domain, the following gives an overview on related work from other software domains.

Auguston's contributions. Auguston suggests an approach to create a model of a program's behaviour [35]. The model is defined as a set of events, the so called event trace. An "event is an abstraction for any detectable action performed during the program execution, such as a statement execution, expression evaluation, procedure call, sending and receiving a message, etc." [35, p. 1]. This implies that events have to be detectable via implementation, e.g., by instrumentation. Each event is defined by a time interval stating the beginning and end of an action. In addition to the event trace, two binary relations over events, precedence and inclusion, are introduced to express temporal relationships. Precedence expresses a sequential order of two events and inclusion states that events are contained within another composite event, e.g., statement execution events within a subroutine call event. Based on this model of a program's behaviour, the event trace can be checked regarding patterns which capture structural and contextual conditions.

In conclusion, the introduced model is basically an event trace format that is optimised to check against assertion rules. Because no explicit assertion rules are given that allow reasoning, e.g., to determine task priorities, the presented approach does

not create added value to this thesis.

Murphy's contributions. The software reflexion model technique by Murphy [36, 37] is another research which is related to the scope of this thesis. The authors present an approach to get an understanding of the system's source code and to gain insights into the system's behaviour. The approach which consists of five steps, derives and iteratively refines a so-called reflexion model. The first step creates a high-level model that describes aspects of the system's structure, e.g., the data flow architecture by reviewing artefacts such as source code, documentation, or expert interviews. Following this, a source model which contains information on the system's interaction including the call graph is created from source code. A map describing the relation between the high-level model and the source model is established next. In contrast to the previous model, the "map is produced manually" [36, p. 3]. Finally, the actual reflexion model is computed to comprehend interactions in the source code from the viewpoint of the high-level model. This is done by merging the information from the source model with that of the high-level model using the map. The resulting reflexion model is then checked and if necessary refined. This step is repeated until the structural information has reached the required level of detailed.

Although this technique describes a list of consecutive steps, the required information for all models except that from source code has to be gathered manually. Furthermore, no algorithms are presented for the one step that can be automated. In conclusion, the presented approach describes a general process of how to proceed in order to get a model that covers the design and implementation of a software system.

Van Hoorn's contributions. Van Hoorn et al. [38] present a framework for monitoring and analysing the runtime behaviour of Java, .Net, and COM-based systems. The framework, called Kieker, requires the instrumentation of the software system. This imposes an "average overhead at below 10 %" [38, p. 2] regarding performance measures such as response times. Monitoring records containing timing, control-flow, and session information, CPU and resource utilization, and memory/swap usage are either stored in the file system, in SQL databases or streamed in-memory. Thus, the monitored records can be analysed online or offline, i.e., during runtime, resp., afterwards. The produced outputs of an analysis are either textual or graphical

visualisations including sequence diagrams, call trees, and dependency graphs.

Although the framework provides comprehensive analysis capabilities including that of the timing behaviour, the results of the individual analyses stand on their own. However, in order to reason about the system as a whole and to reconstruct an architectural model covering the entire system behaviour as targeted within the scope of this thesis, a solution would have to be developed on top of the individual analyses.

Krogmann's Contributions. Krogmann et al. [5, 8] introduce an approach for reverse engineering a behaviour model from Java byte code using genetic search. The goal is to get a representation of the system's components which can be used for performance predictions. The approach is based on static and dynamic analysis of byte code. Via the instrumentation of byte code and following execution in a previously defined test bed that is explicitly designed for the application, the amount of executed byte code and the inputs and outputs of a component are recorded individually. A genetic search is then used on the monitored data in order to discover functional dependencies in the control and data flow. The resulting parametrised model of a component's behaviour consists on the one hand of estimations on how many byte code instructions are executed depending on the input parameters and on the other hand of mathematical formulas describing the number of external calls and their relationship regarding a component's input. Finally, the byte code is executed on the target platform to determine timing values for individual components. All those pieces of information enable then a performance prediction of the entire system.

The presented approach "focuses on business information systems" [8, p. 13] and has to be extended to work with embedded systems. In contrast to what is targeted within the scope of this thesis, however, this approach tries to capture the internal program logic of the system under investigation. This endeavour is rated as legally critical and not targeted within this thesis in order to protect the intellectual property. For that reason black-boxes like software functions are just approximated and not further analysed.

2.3. Other Domains

Similar problems to those discussed within the scope of this thesis can also be found in other domains. One of these domains is the automatic modelling of work flow processes, which is also known as work flow mining. The goal is to generate a formal

model of management steps in an automatic manner. However, there are two main differences. On the one hand, process mining misses a notion of time. On the other hand, the algorithms need multiple traces as input in order to gain confidence in a solution.

Cook's contributions. Cook et al. compare in [39] three different methods for sequential process discovery. Each method generates as a result a finite state machine that covers all behavioural aspects of a process. The first method is called KTAIL and works purely algorithmic. It origins from a string algorithm which looks for prefixes that have the same set of tails for a given length. Those prefixes are then combined to equivalent classes. This has the effect that loops in the trace are unrolled unless they are merged afterwards. A drawback of this method is, that it is not robust in the presence of noise. The seconds methods that is presented is purely statistical. RNET is based on a neural network which is trained using a window of the event stream with a specified length. The last method is called MARKOV and combines both an algorithmic and statistical approach. There, the probability of event sequence in the trace are determined with the help of a Markov model.

In [40], the idea behind the methods above is extended in order to capture and model also concurrent behaviour. As a result of this, the method generates Petri nets instead of finite state machines. The methods works by determining the metrics entropy, event type count, periodicity, and causality from the trace. Based on these metrics, the method determines when concurrent behaviour is occurring and what dependence relationships, such as fork and join, exist. Finally, dependencies are inferred with the help of statistical and probabilistic analyses so that as much of the available event stream as possible is explained.

Besides the fact that the used event traces have no notion of time, the methods also require guidance from someone familiar with the underlying process who tunes the parameters accordingly. Furthermore, the resulting models are just intended as initial models of the process and have to be refined manually afterwards.

Van der Aalst's contributions. Van der Aalst et al. [41] introduce an algorithm which extracts a process model from an event log. The log contains information about the workflow process as it is actually being executed. This means that well-defined steps in the workflow, the events, are recorded sequentially and with every execution a new sequence is created. Then, the presented algorithm analyses such a log and cre-

ates a corresponding Petri net. To give the method a notion of time event types such as task start, task complete, task withdraw, and task resume can be added. However, no noise, e.g., false records must be contained in the event log. Based on the recorded event sequences, the algorithm constructs places with connecting transitions in the Petri net, if it detects a causal relation between two transitions in the log.

Besides the stated limitations of the algorithms that they cannot deal with short loops and that they derive behaviourally equivalent models instead of rediscovering the same model, the algorithm requires multiple event sequences to work. So, this algorithm can be seen as an alternative way to derive the call graph within a process. However, this isn't the main challenge tacked by this work. Nevertheless, there is still the problem that no task interactions, e.g., due to scheduling or synchronisation mechanisms, are considered.

Wen's contributions. Wen et al. introduce in [42] a novel approach for process mining based on event types. For this purpose, the authors introduce two events types called START and COMPLETE that break up the atomicity of tasks. In that way, the fact that tasks need time to execute is exploited and parallelism can be detected. The proposed algorithm is an extension of an existing algorithm which takes the newly exposed temporal information into consideration. At first, the causality information is used to derive ordering relations between individual tasks. Based on this information, the algorithm generates a Petri net.

Although this approach extends existing ones by a notion of time, timing information such as the timespan of the tasks or their recurring appearances are not subject of the process mining.

3

Software Development for Multi-core Architecture

The automotive industry is constantly facing new demands. Modern cars have to produce lower pollution and become more energy efficient in order to comply with more restrictive environmental laws. Another major goal is to reduce the number of accidents, which is a result of the increasing individual transport by improving the active and passive safety in vehicles. Moreover, the demand for comfort and assistance features rises drastically. All these developments and improvements lead to an increasing system complexity.

Most vehicle features are implemented in dedicated electronic control units (ECUs). This means that there is, e.g., one ECU for the engine management and one for the steering. Figure 3.1 gives an impression of the resulting complexity by visual-

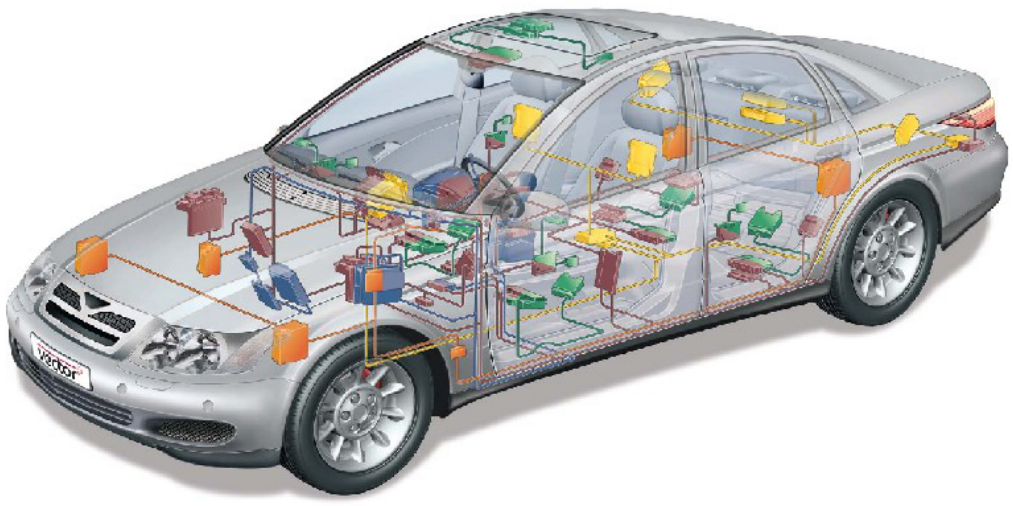


Figure 3.1.: Schematic Overview of ECUs in a Luxury Car from Year 2002. Copyright: Vector Informatik GmbH.

ising the 76 individual ECUs of a luxury car 15 years ago. Nowadays, this amount is reached by mid-range vehicles and modern luxury cars are expected to be equipped with more than 100 ECUs.

3.1. Target Mapping

The software development process for automotive and other embedded systems includes a step called target mapping. In this activity, artefacts which result from prior steps of the development process and which are independent of the target platform, such as functional models, are adapted to the target platform. Thereby, a multitude of different system characteristics have to be considered, e.g., the available hardware platform including the processing units and their connection to the periphery or the operating system (OS), which manages the temporal scheduling in a multi-tasking system.

Multi-core architectures pose an additional challenge to the target mapping and increase the complexity of the systems under development further. There is a multitude of possibilities to distribute tasks on the available processing units. At the same time, the correct execution of the software has to be guaranteed which is achieved by sticking with the defined execution order of functions. But this order is initially designed for a sequential execution and has to be paralleled by the target mapping in order to benefit from the multi-core architecture. However, the mapping of a task to a processing unit has a crucial influence on the system's timing behaviour, its performance and safety. As a consequence, the goal of the target mapping is to optimise these system qualities without changing the execution behaviour.

Parallelism can be performed on different levels and starts with the problem itself, e.g., a feedback loop. To solve this problem, a *parallel algorithm* can be applied instead of an algorithm that works sequential. In a next step, the implementation of the algorithm is divided into individual parts, the functions, which are called runnable in automotive terminology, so that they can run in parallel in a computer program. A possible realisation of such a *parallel program* is the execution in multiple independent processes, the so-called multi-tasking. Finally, *parallel computing* requires that the execution of the program is running concurrently on the available multi-core hardware platform.

The development step of adapting software to the target platform, which consists of many phases, realises parallel program and parallel computing. A possible real-

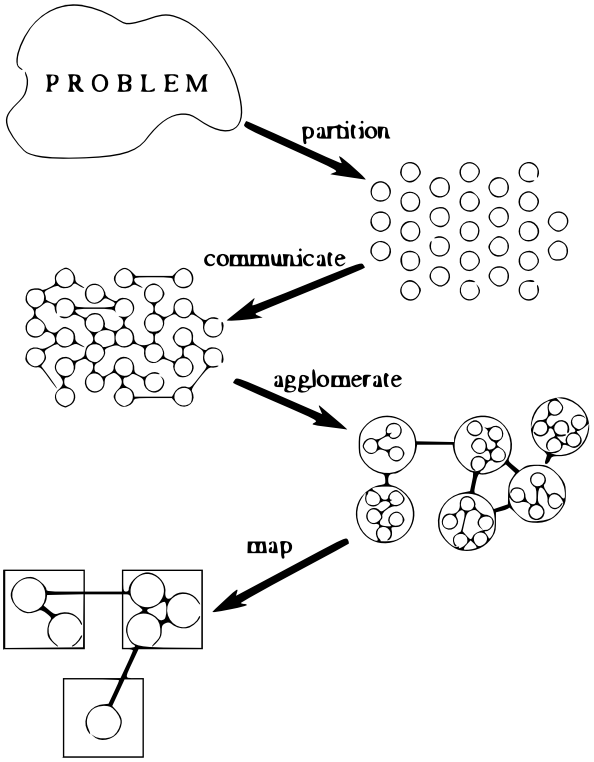


Figure 3.2.: PCAM Method as originally introduced by Foster. Reprinted from [46].

isation of the target mapping is, e.g., the PCAM Method as originally introduced by Foster [46]. As depicted in Fig. 3.2, it consists of the following four phases, which are discussed in detail in the following sections:

- **Partitioning:** The mapping of functions to tasks.
- **Communication:** Analysis of the requirements on the communication between tasks on the available multi-core platform.
- **Agglomeration:** Optimisation of the task structure with the goal to minimise the overhead by communication and synchronisation.
- **Mapping:** The mapping of the optimised tasks to the different processing units considering the available hardware properties such as the speed of the memory connections.

3.1.1. Partitioning

The partitioning represents the first phase towards a parallel execution. It applies the design paradigm “divide and conquer”, which states that a problem has to be divided in independent parts. Then, these parts have to be solved in parallel. The goal of

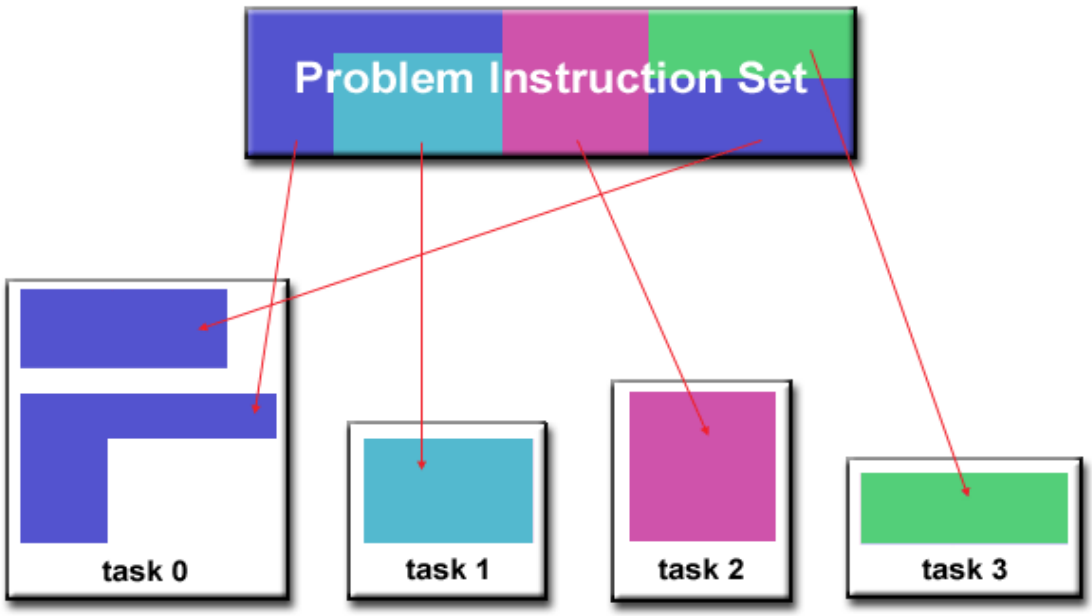


Figure 3.3.: Functional Decomposition. Reprinted from [47].

the partitioning is to create a large number of small functions that do not contain duplicate data and computations. This means that the partitioning does not only consider the computations for solving the problem but also the data on which the computations are executed.

There are two general approaches for partitioning available [47]: domain decomposition and functional decomposition. In the domain decomposition, at first the data of the system are divided into equal sized parts and then all computations that are producing these data parts are packed together in a task. In opposite to this, the functional decomposition splits the system according to their functionality and then these functions are bundled together with the required data into tasks.

For the development of hard real-time systems in the automotive industry, mainly the approach of functional decomposition is applied in which all the runnables are bundled to tasks according their activation requirements. The activation requirement of each runnable is defined, e.g., by the feedback loop it implements or the sampling rate of a sensor. Fig. 3.3 depicts this approach, in which each colour defines a different recurrence period.

3.1.2. Communication

The goal of this phase is to determine the correct position of a runnable within a task. Fig. 3.4 depicts the data flow between six runnables in a task.

This results in a directed graph that contains often a cycle. To achieve a sequential execution order for the runnables, which is required for each task, it is necessary to break up this graph by eliminating all cycles within the graph. At the same time, the correct real-time behaviour of the software functions on the target platform has to be preserved.

To achieve this, the requirements on the dynamic behaviour are clearly denoted for each runnable using *execution order constraints* and *age constraints*. An execution order constraint defines the relation between runnables in such a way that one runnable has to be executed before the other, e.g., because of a data dependency. If a runnable does not have any dependency to any other runnable regarding its execution order, it can be placed arbitrarily within the task. An age constraint defines the maximum age of a data value, i.e., the maximum time since a value of the data signal was produced last. This guarantees that the runnables work with the latest data. Fig. 3.5 extends the example introduced in Fig. 3.4 by the annotated timing constraints.

With the help of the timing constraints, it is possible to eliminate cycles within the graph and to achieve a sequential execution order for the runnables. Fig. 3.6 shows a possible execution order for our prior introduced example.

In Fig. 3.6, the runnables are positioned within the task so that all defined execution order constraints are met and that the times between the production and con-

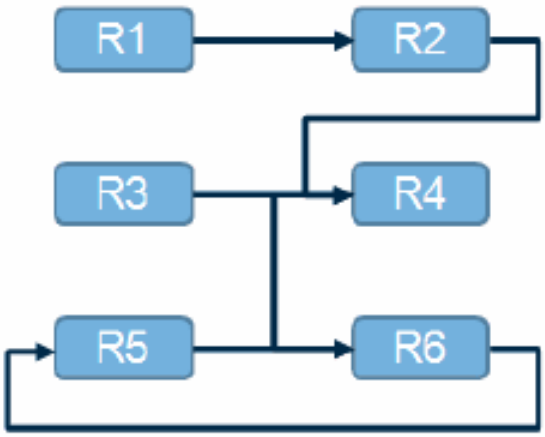


Figure 3.4.: Data Flow between Runnables. Data Flow between the Runnables (R1 – R6). Adapted from [48].

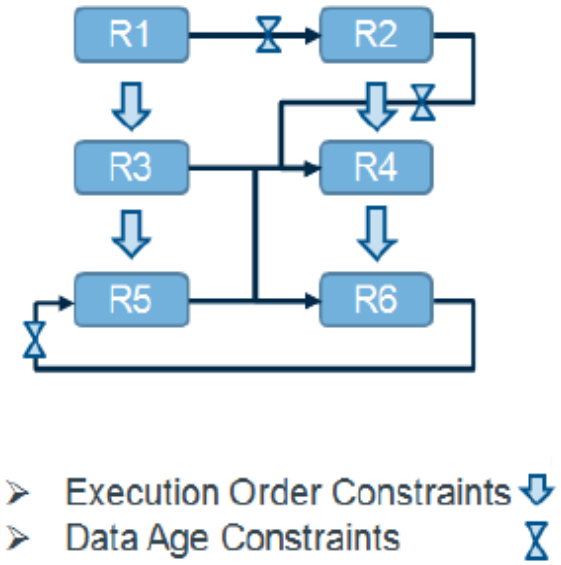


Figure 3.5.: Timing Constraints to annotate Data Flow. Data flow between the runnables (R1 – R6) with annotated timing constraints. Adapted from [48].

suming of the data, which are constrained by an age constraint, are minimised. However, the presented execution order is just one of perhaps many possible solutions. Although the solution still contains cyclic data dependencies, these data accesses are not crucial for a correct behaviour because no maximum data age is defined. This means that in these cases it does not matter whether a runnable works on the latest data values or on data which has been produced during the previous task execution.

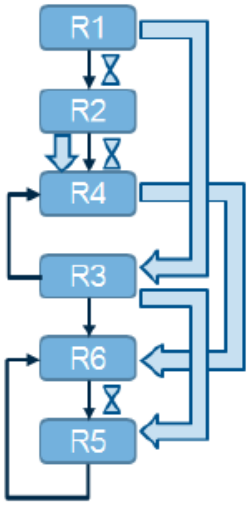


Figure 3.6.: Execution Order of Runnables. Possible execution order of the runnables (R1 – R6) within the task. Adapted from [48].

Nevertheless, also the fact that usually not all requirements are known at the beginning of the target mapping has to be considered. This is a crucial problem especially in the automotive industry, where a distributed software development is practised, i.e., parts of the software are developed by the OEM and other parts are developed by the Tier-1s. For this reason, all the steps of the target mapping have to be performed again with every integration of a new software version.

3.1.3. Agglomeration

The transition between the analysis of the communication and the subsequent optimisation of the task structure is fluent and is also performed iteratively. While the previous step focuses on the communication between the runnables within a task, this step considers the interaction of tasks and the resulting impact on the communication. Thereby, it is necessary to protect the data signals which are processed by the runnables from concurrent access in order to ensure *data stability* and *data coherency*.

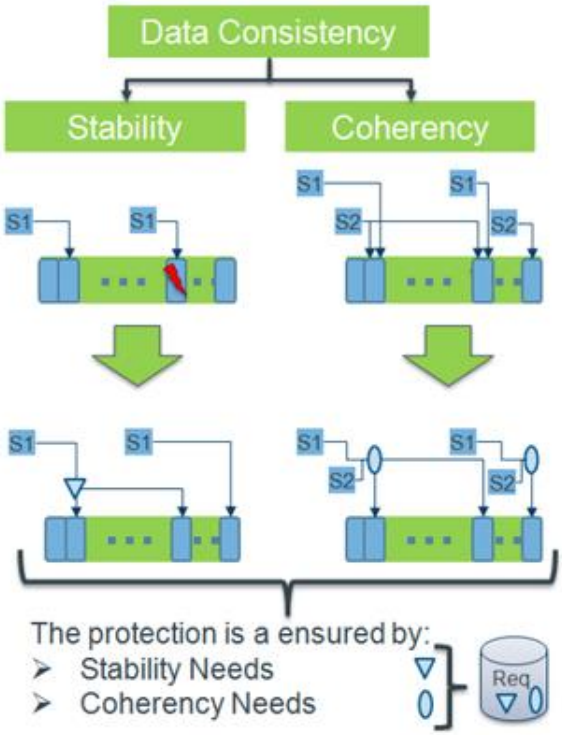


Figure 3.7.: Need for Data Stability and Coherency. Example showing the need for data stability and coherency. Reprinted from [48].

Data stability As soon as runnables are executed in parallel and data is exchanged, it is very likely that a data value is modified although a different runnable is still in need of this value. If the data is scalar and can be written atomically, then the value of the data signal cannot be corrupted by a concurrent write access. This means that a read access to a data signal yields either the old or the new value. But as soon as there are multiple read accesses, either within the same runnable or across multiple runnables, and each time the same value is expected, then it can happen that the stability of the data signal is lost. This situation is depicted on the left side of Fig. 3.7, in which a data signal $S1$ has to be stable across multiple runnables.

In some cases, a concurrent modification of data is acceptable because the values are used in encapsulated parts of the algorithm or because the data features low dynamics and thus, can change only slightly. But in other cases, changing a value such as a boolean value or the state variable of a state machine during a computation has a severe impact. To avoid this behaviour, the need for stability of a data signal is defined first. Then, this data stability is realised in the agglomeration phase, e.g., via buffering.

Data coherency A change in well-defined data structures such as structs or arrays while a runnable is still processing this data can also have a severe impact. For example, two pieces of information such as a flag and its complement have to be written and read in a coherent way because otherwise it can happen that the data is read and in the meantime a part of the data set is already modified. This case is visualised on the right side of Fig. 3.7, in which coherency for the data signals $S1$ and $S2$ is required.

In general, this problem appears on communication between tasks that have different activation frequencies. Thus, it has to be defined for each of these data signals, whether a data coherency is needed. In the agglomeration phase, the coherency is then realised by implementing critical sections for the affected data, e.g, via semaphores.

3.1.4. Mapping

Finally, the mapping realises the allocation of the defined tasks to the available processing units of the target platform. Tasks, which are executed on the same processing unit access the same local memory. But if tasks communicate between processing

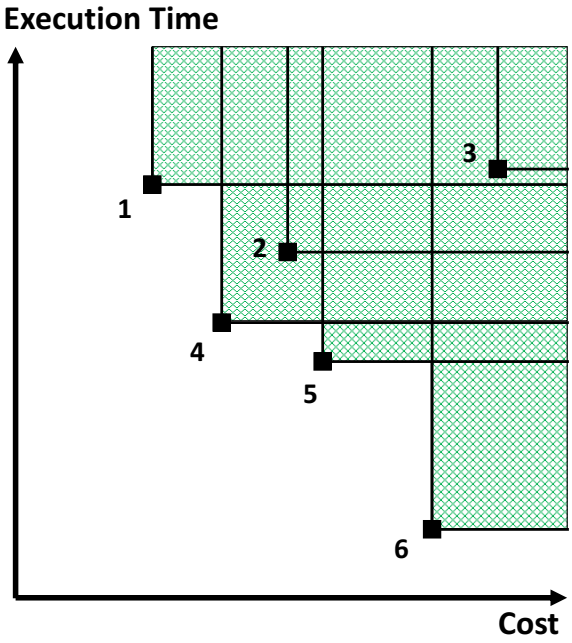


Figure 3.8.: Pareto-optimal Solutions. Solutions compared regarding their minimum execution time and minimum cost. The hatched areas visualise the areas in which a solution is pareto-optimal.

units, then data has to be transferred between these processing units. In doing so, a temporal overhead arises which results in an execution delay. In the worst cases, this overhead becomes so massive that the results of the previous phases communication and agglomeration are invalidated because the real-time behaviour of the software functions cannot be guaranteed any more. Thus, the mapping considers mainly the resulting communication overhead of the tasks. This challenge is even bigger for heterogeneous processor architectures, where not all processing units are equal but feature, e.g., different frequencies or memory interfaces.

There are multiple possible goals that can be targeted by the mapping, e.g., an equally distributed load over all processing units, the so-called load balancing. The underlying problem of the mapping is equal to that of bin-packing. This means that there are a multitude of possible solutions. Thus, the goal of the mapping is to consider the most comprehensive subset of the possible solution space in a reasonable amount of time in order to find a so-called pareto-optimal trade-off. This is necessary because the objectives of the partitioning and mapping phases oppose each other, e.g., real-time properties vs. load balancing. A result of the mapping is a pareto-optimum and thus, superior to other solutions, if it performs better in at least one objective than all other solutions and if it is at least equal in all other objectives at the same

time.

Fig. 3.8 visualises the problem of pareto-optimal solutions. In this example, the best trade-off between the two objectives minimum execution time and minimum cost has to be found. The figure shows six solutions within the design space and the areas in which they are pareto-optimal. Thus, there are four pareto-optimal solutions (1, 4, 5, and 6) in this example from which the "best" has to be chosen then.

3.2. Timing Simulation

Timing simulation is an approach for examining temporal properties of a real-time system as shown in Fig. 3.9. Established techniques for analysing whether a system is capable of meeting all the required deadlines such as timing analysis or scheduling analysis cannot handle the exploding complexity in automotive systems and are more and more replaced by timing simulation. The biggest advantage of employing timing simulation during the development is that it allows one to make design decisions in an easy and time efficient way. For example, an engineer can analyse the impact of changing the sequence of runnables within a task during the target mapping by simulating the system behaviour and determining the resulting communication overhead. By making assumptions, this is even possible, if the target platform is not physically available or if the software system is only partially known. With progressing development, more information on the system under development is available and thus, less assumptions have to be made so that the simulation results get closer to the system's actual behaviour.

The realisation of a timing simulation can be based on, e.g., a discrete-event simulation [49]. In a discrete-event simulation, changes to the system, so called events, can only occur to a particular instant in time. Between consecutive events, no change in the system is assumed to occur, which allows the simulation to directly jump in time from one event to the next instead of continuously updating the system's internal state in small time slices. Based on the occurred events, the system behaviour during runtime can be comprehended and the performance determined.

In case of an underlying discrete event simulation, the timing simulation works on an abstraction of the system, which consists of models for the hardware, software and operating system. Each model is represented by a state machine which describes the system's behaviour and the input/output interfaces that are required to interact with other model parts. A transition from one state to another is caused by interactions

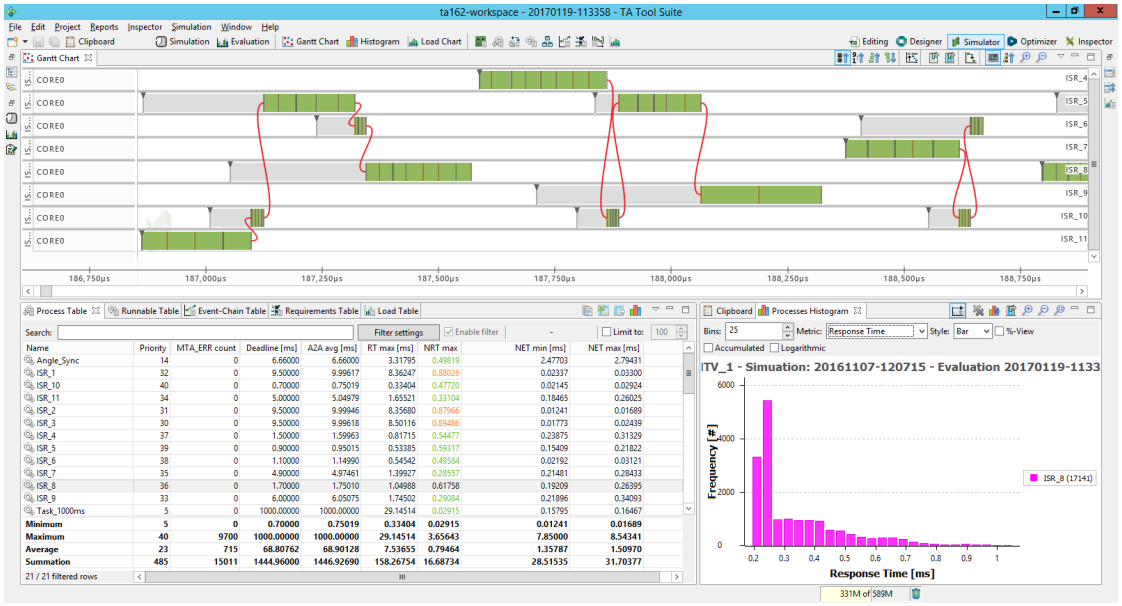


Figure 3.9.: Screenshot of Simulation Results in TA Simulator. Screenshot of simulation results as presented by the commercial timing simulation tool TA Simulator[1]. The upper half visualises the events in a Gantt chart. The lower half shows the performance metric values calculated from the events in a table and their distribution in a histogram.

via events. An event consists of a time stamp, which defines the occurrence moment of the event, a signal, which is necessary to differ between the interactions of two model parts, and a value, which gives additional information on the signal, e.g., an identifier and an event counter. For a chronologically correct sequence of events, the simulation holds a queue of all occurred events and sorts them according their time stamp. During the execution of the simulation, the global time is set to the time stamp of the next event in the queue and the transitions for the affected state machines are triggered.

The employment of a discrete event simulation makes it possible to model the system in a probabilistic manner. This has the effect that with each execution the parameters of the state machines are varied within their valid range so that their stated statistical estimators are met. Because this approach covers the complete range of the model parameters, scheduling anomalies can be exposed and an extensive coverage of the exploration space can be achieved. The result of the timing simulation is a sequence of events, a so-called trace recording, which contains all state transitions of the system during execution. These events are then used to determine performance

[1] `http://www.timing-architects.com`

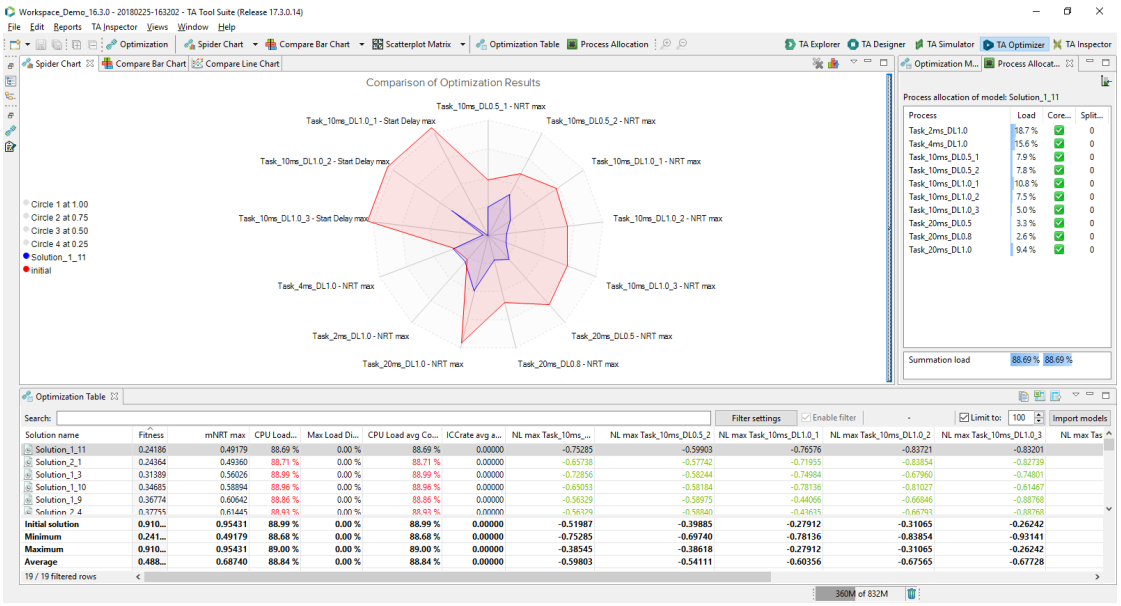


Figure 3.10.: Screenshot of Optimisation Results in TA Optimizer. Screenshot of optimisation results as presented by the commercial tool TA Optimizer[2]. The upper half opposes the fitness of the initial model (red) and that of an optimised model (blue) in a radar chart. The lower half shows the individual metric values considered by the fitness in a table.

metrics of the real-time system, such as the response times of tasks of communication overhead that allows one to make the required design decisions.

3.3. Model-based Optimisation

Another approach which gained in importance with the arrival of multi-core architectures in the automotive development is the model-based optimisation. By applying model-based development, e.g., via the AUTOSAR methodology, it is possible to systematically evaluate different variants of the system under development in order to maximise system performance according to user-defined metrics such as utilisation or inter-core communication.

For example, the huge design space of the target mapping for multi-core architectures can be efficiently evaluated by applying a genetic algorithm on a system model. There, specific parts of the system model such as a task's priority or the task-to-core-allocation are altered randomly by mutation and crossover algorithms. The performance of the resulting model, the so-called fitness, is then determined by employing a

[2] http://www.timing-architects.com

timing simulation or timing analysis. The fitness is a scalar value, which is calculated as a weighted sum of user-defined metrics and defines the optimisation goal. According the theory of the survival-of-the-fittest, the models which are evaluated best are selected to continue with the genetic algorithm until a termination condition such as a fixed number of generated models is reached. As shown in Fig. 3.10, the result of the optimisation consists of a pre-defined number of system models, the so-called final population, which have the highest fitness values during the optimisation process. Based on this final population, the engineer can choose a pareto-optimal solution for the system under development.

4

Real-time Automotive Model

Most ECUs in a vehicle represent real-time systems. There, the correctness of a system's behaviour depends not only on functional correctness but also on temporal accuracy:

Definition 4.1 (Real-time System)**.** "A real-time system is a system that is required to react to stimuli from the environment [...] within time intervals dictated by the environment" [50, p. 2].

This consideration of timing behaviour is an extra challenge which requires substantial effort, but it is essential to the development process. For example, timing simulation is applied to examine temporal properties of real-time systems by use of discrete-event simulation. In a discrete-event simulation changes to the system, so called events, can occur only at a particular instant in time. Between consecutive events no change in the system is assumed to occur, which allows the simulation to directly jump in time from one event to the next instead of continuously updating the system's internal state in small time slices. Because events can only occur at a particular instant in time, a discrete-event simulation can work with an abstract description of the system under development. Thus, the most abstract description of a real-time system can be defined according to [49, p. 11] as follows:

Definition 4.2 (Real-time System Model)**.** A real-time system $S = (\tau, \Pi, \xi)$ consists of a set of processes τ, a set of processing units Π, and a set of schedulers ξ.

A set of processes represents the dynamic application software and defines the execution demand to the system. In contrast to this, the processing resource specifies the amount of execution performance that can be provided by the system. Finally, the scheduler establishes a connection between these two by describing how available resources are utilised in order to cope with arising execution demands.

With every piece of information added to the model, the abstraction is reduced and the simulation accomplishes more accuracy. This goes as far as to simulate the

timing behaviour in a cycle exact manner, which however, would go beyond a reasonable amount of time for simulation. Thus, a trade-off between simulation runtime and level of detail described in a model has to be found. A detailed description on what is covered by each part and which details can be added to achieve more accurate simulation results is discussed in-depth in the following.

4.1. Processing Units

The hardware model focuses on the description of the available computing power. It represents the balance to the software part, which makes a demand on the required performance. The execution capabilities in an embedded system are provided by the ECU. An ECU includes at least one micro-controller plus peripherals such as timers and analogue ,resp., digital inputs and outputs.

The hardware model is deliberately kept elementary, because the focus of this work is on reversely engineering the software rather than on identifying characteristics of the hardware. Although the hardware has an impact on the real-time behaviour of the software, this impact is mainly reflected by the execution demand of the software, e.g., each memory access extends the execution time. Furthermore, this work only takes ECUs into account that contain exactly one micro-controller, which is currently state-of-the-art in the automotive industry.

Definition 4.3 (Hardware Model)**.** The hardware model consists of a single micro-controller Π and defines the processing frequency $\nu(C_y)$ for each of the $n = |\Pi|$ available processing units $C_y \quad \forall y \in \{1, \ldots, n\}$. Each processing unit C_y of the micro-controller is connected to the shared memory, which allows the sharing of data values.

A real-time system that consists of a micro-controller with exactly one processing unit, which implies $|\Pi| = 1$, is called a single-core system. In cases of $|\Pi| > 1$, we speak of multi-core systems. For example, Figure 4.1 depicts a quad-core micro-controller, which means $|\Pi| = 4$.

Each processing unit C_y features a constant processing frequency $\nu(C_y)$ that does not vary in time. Thus, dynamic voltage/frequency scaling (DVFS), which reduces the processor voltage or frequency to minimise power consumption, cannot be represented in the model. However, it is possible to model symmetric multi-core processor systems, in which all processing units of a micro-controller have the same frequency, as well as asymmetric ones. In general, the processing frequency $\nu(C_y)$ denotes, how

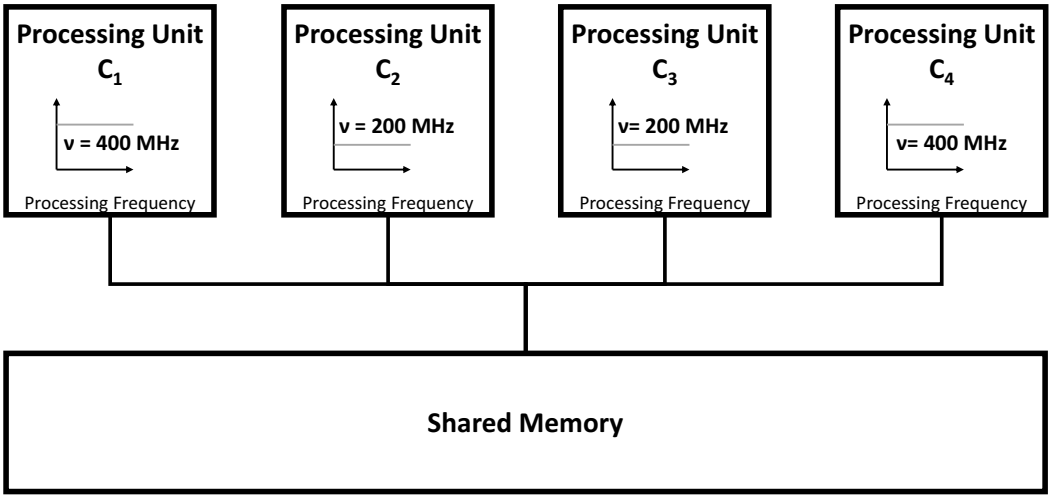


Figure 4.1.: Hardware Model. Schematic overview of the information contained in the hardware model. Shown is a quad-core micro-controller with its four processing units ($C_1 - C_4$). The processing units C_1 and C_4, resp., C_2 and C_3 run at a constant frequency ν of 400 MHz, resp., 200 MHz.

many instructions the processing unit C_y can calculate in a second. Thus, a task's execution time is determined by the number of instructions that has to be executed.

The mechanism, which allows tasks to share data values over processing units, the inter-task communication, is realised in a symmetric way. For this, the micro-controller has a shared memory, which all processing units C_y access in the same way with the same access time. This considers a non-blocking access approach to the shared memory, because blocking mechanisms delay all read and write accesses if there is an ongoing write access to a data signal until this signal has been written. Additional peripherals on an ECU such as communication buses, timers, or further blocking resources are not considered in the model.

4.2. Processes

The software subsystem of a real-time systems is represented by the dynamic application software. It consists of a set of processes and determines the execution demand to the system.

Definition 4.4 (Process Set)**.** Let P_i be a process, then a process set τ =

$\{P_1, \ldots, P_n\} \mid n \in \mathbb{N}_0$ is a collection of all processes in a real-time system S.

As originally published in [49, p. 11], a process can be defined as a transformation of a set of input data $\Phi^I(i)$ to a set of output data $\Phi^O(i)$. This abstraction is visualised in Fig. 4.2. This thesis builds upon this modelling idea and extends it by the additional property priority o_i:

Definition 4.5 (Process)**.** A process $P_i = (a_i, e_i, d_i, o_i, \Phi^I(i), \Phi^O(i))$ is defined by the temporal properties activation stimulus a_i, execution time $e_i(C_y)$, and deadline d_i and by the logical properties priority o_i, input data $\Phi^I(i)$ and output data $\Phi^O(i)$.

Each process is activated by a stimulus a_i. This stimulus can origin from the environment or from the system itself, e.g., due to periodicity or inter-process activation. There is a wide range of patterns available that allow one to model not only periodic but also infrequent activations including jitter or sporadic behaviour. The patterns supported in the developed model are described in detail later on in Sec. 4.2.3.

The execution time $e_i(C_y)$ defines the time that a processing unit C_y takes to execute the process without interruption. Thus, it states a demand on the provided computing power. This demand can be either specified as a duration for each processing unit or abstracted as the number of instructions which have to be processed. The advantage of latter is that it does not have to be determined for each processing unit individually, because the execution time results from the amount of instructions

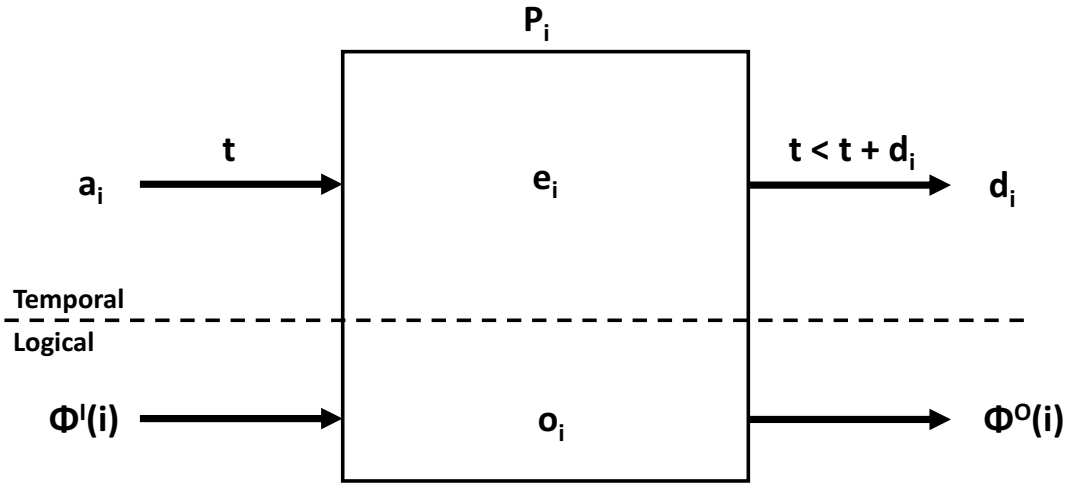


Figure 4.2.: Process Model. Visualisation of an abstract representation of process P_i including its properties activation stimulus a_i, execution time $e_i(C_y)$, deadline d_i, priority o_i, input data $\Phi^I(i)$ and output data $\Phi^O(i)$. Adapted from [49, p. 12].

and the frequency of a processing unit. This is helpful especially in heterogeneous systems.

The deadline d_i and the priority o_i are parameters used for scheduling. Depending on the applied scheduling algorithm, the parameters are considered by the scheduler in order to determine which process is assigned a processing resource next. Considering the classification of the real-time system (hard, firm, soft), the deadline defines the latest moment in time relative to its most recent activation before the process's execution should, resp., has to be finished. In contrast, the priority is an integer which establishes a relative order of importance between processes.

Input data can either come from another process or from the environment of the real-time system and output data can, analogously, go to another process or the environment. In contrast to the model defined in [49, p. 11], the model used in this work cannot only require and provide signals as input, resp., output data but also semaphores and spinlocks. A semaphore is a mechanism for limiting the number of concurrent accesses to a resource. It is defined by its initial value n, which states the maximum number of concurrent accesses. If a semaphore is initialised with $n > 1$, its called a counting semaphore. This is due to the fact that each granted access to a resource, resp., each exit decreases, resp., increases the number of remaining concurrent accesses allowed to that resource. In case of a mutex ($n > 1$), the semaphore realises mutual exclusion. A spinlock is defined as a mechanism which causes a process trying to acquire a lock to simply wait in a loop while repeatedly checking if the lock is available. Thus, both semaphores and spinlocks allow one to control and synchronise communication between processes.

In our model, a process can either emerge as an interrupt service routine (ISR) or a task.

Definition 4.6 (Task)**.** A task $T_i = (a_i, e_i, d_i, o_i, s_i, q_i, \Phi_i^I, \Phi_i^O)$ is defined by the temporal properties activation stimulus a_i, execution time e_i, and deadline d_i and by the logical properties priority o_i, pre-emptability s_i, maximum size of activation queue q_i, input data Φ_i^I and output data Φ_i^O.

In contrast to an ISR, a task requires the additional parameters s_i and q_i. The pre-emptability s_i states, whether a running task can be interrupted by the scheduler. If a task is non pre-emptive, it continues execution until its termination or until a schedule point is reached although a task with a higher priority is ready. Instead, ISRs can be interrupted by ISRs with a higher priority at any time. A detailed description on schedulers and how this parameter is used for scheduling is presented in Sec. 4.3.

The parameter q_i states the maximum size of queued activation requests for a task, i.e., how many instances of a task can be activated and, thus, reside concurrently in the system at any time. A value "1" means that only a single activation is permitted for this task and that it has to finish execution before it can be activated again.

So far, only processes on *Process Level* have been considered, i.e., one sees them as black boxes without any knowledge of what is happening inside. Based on this description, it is already possible to simulate the timing behaviour of processes roughly. However, to get more precise results and to get closer to the actual system behaviour, it is necessary to model tasks as white boxes. This is realised in the developed model via the call graph, which is introduced in detail in the following.

4.2.1. Call Graph

Fig. 4.2 depicts a process as a transformation of a set of input data $\Phi^I(i)$ to a set of output data $\Phi^O(i)$ that takes e_i time to execute. This abstraction lacks information on when or how often the input/output data is accessed or what exactly is happening during the execution time of a process, which is essential to get more precise simulation results. As a consequence, Def. 4.5 is extended to system level by adding a description on how the process is interacting with the system.

This description is provided as a rooted, directed, acyclic graph, the so-called Call Graph:

Definition 4.7 (Call Graph)**.** Let V be a set of system interactions and let $E = \{e_i\}$ be a set of tuples $e_i = (v_k, v_l)$ that link two interactions $v_k, v_l \in V$ in such a way that the graph is free of cycles and all sequences origin in a single interaction. Then the Call Graph G is defined as $G = (V, E)$.

With each execution of a process a sequence of system interactions is performed. The call graph describes all possible sequences which can occur during execution in a single graph. In the developed model, a process can perform the following system interactions:

Runnable Call: A task calls the functionality of a runnable within its context. Runnables are not only responsible for consuming and producing the input, resp., output data of a task, but are also main contributors to the execution time of a task as described in Sec. 4.2.2.

Mode Switch: A mode switch represents a fork from which on different sequences of interactions can be observed, e.g., during initialisation of the system. In the

developed model, a sequence connected to the fork can be selected either in a deterministic manner according specific values of the input data or randomly. Latter, allows one to express a mode switch in a probabilistic manner, i.e., one branch is, for example, taken in 70 % of the cases and the other one in 30 % of the cases.

Schedule Point (not allowed in ISRs): A schedule point explicitly calls the scheduler. It marks a deliberately chosen point in time and divides a task into so-called schedule sections. A task which is defined as non pre-emptive can only be pre-empted between these schedule sections.

OS Events (not allowed in ISRs): OS events provide a mechanism for synchronisation and can be set, cleared and waited for by a task. A task has to wait for a specified event until it is set. Once the event occurs, the task's execution can continue.

Inter-process Activation (not allowed in ISRs): A task can activate another task at any time during its execution by triggering a stimulus.

The call graph differs from a typical control flow graph that states the order in which individual statements, instructions or function calls of an imperative program are executed, in such a way that only a small subset of interactions are allowed on process level. Its main purpose is to define the sequence of runnables that are called in a process. If not all information is available to do so, the call graph allows one to model the order in a probabilistic manner, as shown by the activity diagram in

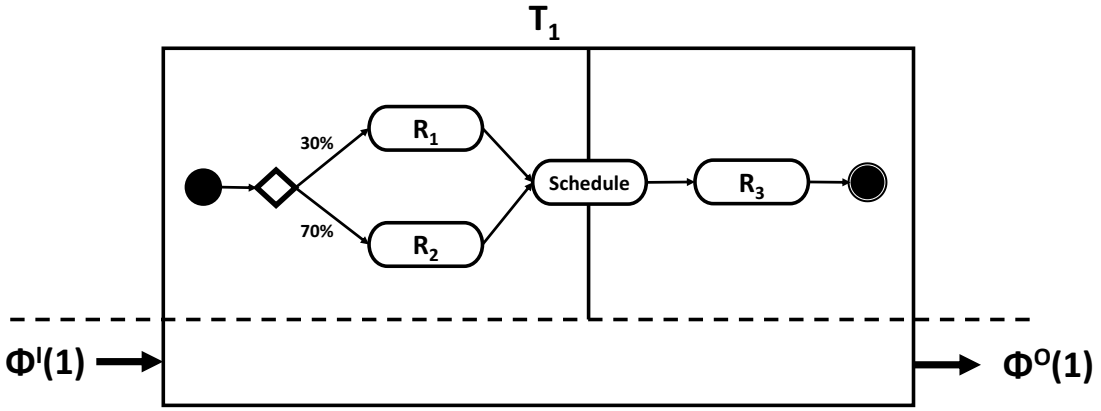


Figure 4.3.: Call Graph. Activity diagram visualising the call graph of task T_1. At first, $Task_1$ calls in 30 % of the cases Runnable R_1 and in 70 % of the cases R_2. After that, the scheduler is invoked via a schedule point and runnable R_3 is called before the task terminates.

Fig. 4.3. The actual program logic, instead, is represented by the runnables, which are discussed in the following.

4.2.2. Runnable

The source code of a task itself does not contribute any functionality to the system. Instead, the functionality of a system is usually implemented by a function. In automotive domain, these functions are called runnable entities or short runnables. They are part of a software component and can be executed and scheduled independently from other runnables.

Thus, a runnable is defined by a sequence of instructions which are executed within the context of a process. The developed model seizes upon this idea and describes a runnable in the following way:

Definition 4.8 (Runnable). Let $\Phi^I(i,j) \subset \Phi^I_(i) \cup \varnothing$ be some input data, $\Phi^O(i,j) \subset \Phi^I(i) \cup \varnothing$ some output data of the process P_i and let $X = (x_0, \ldots, x_n)$ define a sequence of instructions that work on this data, then a runnable R_j is defined as $R_j = (\Phi^I(i,j), \Phi^O(i,j), X)$.

A process calls the functionality of a runnable within its context. This means that the runnables are responsible for transforming the input data of a process $\Phi^I(i)$ to its $\Phi^O(i)$. Each runnable can also produce and consume local data signals. These are additional signals that are neither input nor output data of the process but rather serve for inter-runnable communication.

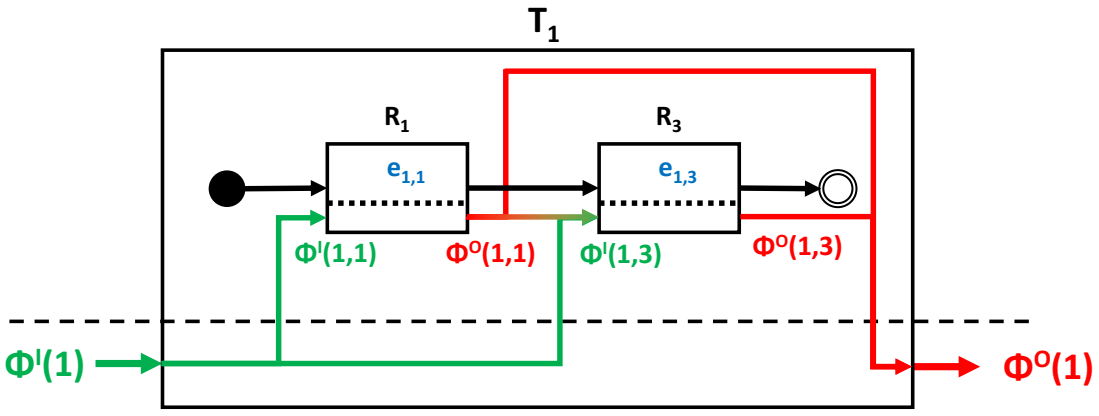


Figure 4.4.: Input–process–output (IPO) Model of a Process. Abstract representation of how the runnables R_1 and R_3 are consuming and producing the input, resp., output data of task T_1.

As shown in Fig. 4.4, runnables are not only responsible for consuming and producing the input, resp., output data of a process, but are also main contributors to the execution time of a process. For these purposes, the developed model provides the following instructions that can be used within a runnable:

Instructions Block: An instructions block states a specific demand on the provided computing power by defining a number of instructions that have to be processed. The execution of this amount of instructions by a processing unit contributes, then, to the execution time of a process. Because the purpose of a model is not to describe the actual amount of instructions exactly but rather in an abstract manner, the developed model provides a wide range of probability distributions that allow one to describe different scenarios, e.g., average or corner cases:

- **Constant:** The amount of instructions is defined by a constant value as shown in Fig. 4.5. This representation is preferred in situations where no further details on the dynamic behaviour of the system are known.

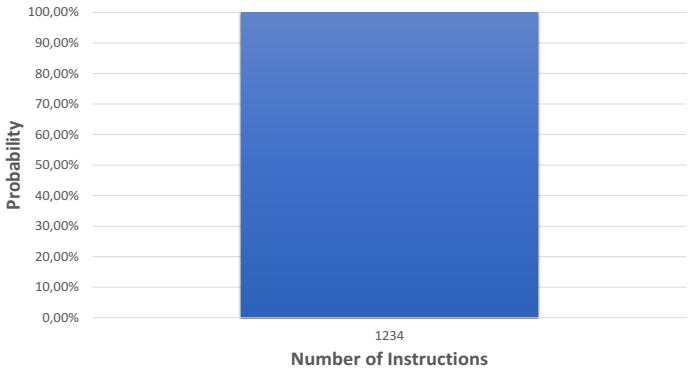


Figure 4.5.: Constant Value. Probability mass function showing a constant number of instructions.

- **Uniform Distribution:** The amount of instructions follows a uniform distribution, which is defined by the minimum and maximum value as shown in Fig. 4.6. This representations is used in cases where the engineer knows that the dynamic behaviour varies between a minimum and a maximum value but the distribution of the individual values is still unknown and for that reason taken as uniform distributed.

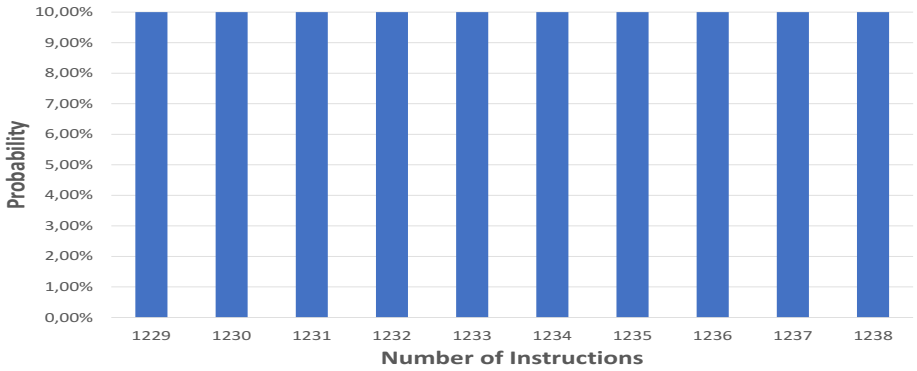


Figure 4.6.: Uniform Distribution. Probability mass function showing that the number of instructions is uniformly distributed between 1229 and 1238 instructions.

- **Normal/Gauss Distribution:** The amount of instructions follows a Normal/Gaussian distribution, which is defined by the average amount, the standard deviation value, and the lower and upper bound as shown in Fig. 4.7. It is used in cases where the amount of instructions is mostly a specific value, the average, but values can vary slightly up to a limit.

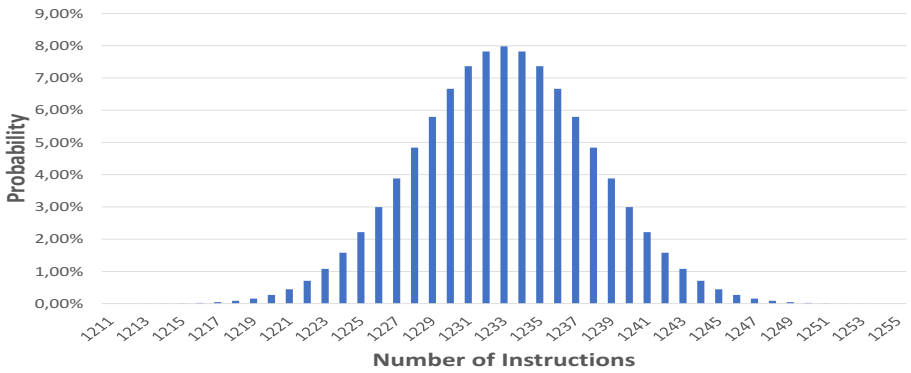


Figure 4.7.: Normal/Gauss Distribution. Probability mass function showing that the number of instructions is Gauss distributed between 1184 and 1282 instructions with an average of 1233 instructions and a standard deviation of 5.

- **Weibull Distribution:** The amount of instructions follows a Weibull distribution [51], which is defined by the shape parameter k, the scale parameter λ, and the lower and upper bound as shown in Fig. 4.8. This representation is mainly used to model situations in which the amount of instructions is close to either the set minimum or maximum.

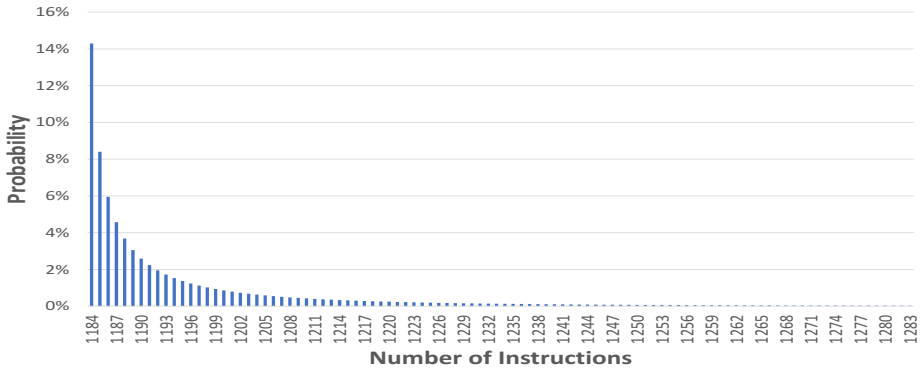


Figure 4.8.: Weibull Distribution. Probability mass function showing that the number of instructions is Weibull distributed between 1184 and 1282 instructions with the shape parameter set to 0.5 and the scale parameter to 5.

- **Beta Distribution:** The amount of instructions follows a Beta distribution [52], which is defined by the minimum and maximum value, and the shape parameter α and β as shown in Fig. 4.9. This distribution allows one to model corner cases, i.e., scenarios in which the amount of instructions is close to both the set minimum and maximum and only a few outliers are between these peaks.

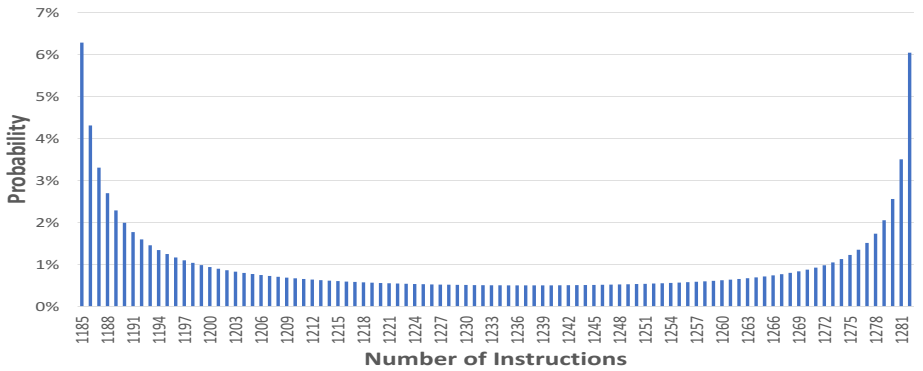


Figure 4.9.: Beta Distribution. Probability mass function showing that the number of instructions is beta distributed between 1185 and 1282 instructions with the shape parameters α set to 0.05 and β to 0.2.

Data Access: A data access defines a read or write access to a data signal. A write access represents the change of a data signal and a read access indicates that the behaviour of the runnable is influenced by the value of that access. Depending on the level of abstraction, a data write access is either stated without any additional information or with the exact data that is written. The former is usually done, if the model is represented in a probabilistic manner and no exact information on

the data flow is known.

Semaphore Access: A semaphore access represents an access to a semaphore or a spinlock and can either indicate a request or release. In case the semaphore and spinlock has reached its maximum, a request results in the blocking of the task in which context the runnable is executed. Instead, a release signals that an additional access to the shared resource is granted.

Runnable Call: A runnable call states the activation of another runnable and represents a sub-function call. The called runnable is then executed in the context of the task in which the calling runnable is executed.

Although a runnable represents the implemented program logic, the model describes this behaviour in an abstract manner. The goal is not to model the source code one-by-one, but to roughly describe what a runnable does and when. For that reason, there is no loop or other control flow statements available, just a plain sequence. However, instructions such as an instruction block can be repeated within a sequence. That way it is not only possible to position data accesses at the correct point in time, but also to model loops by repeating each sequence within a loop for the given number of loop cycles.

4.2.3. Stimulation

A stimulus is required to activate processes and represents responses to changes in the environment or the system itself. These reactions can appear regularly or in a random manner. For that reason, different stimulation patterns are available in the developed model that allow one to describe a wide range of activation behaviour.

4.2.3.1. Inter-Process

The inter-process stimulation is defined as a process activation that is triggered during the execution of another process. Thus, it does not follow a specific temporal pattern, i.e., every millisecond, but rather represents a causal relationship between two processes.

Consider, for example, two processes P_2 and P_3 that calculate some values and a third process P_1 that uses these values as input. Then, an efficient way of processing is that both processes P_2 and P_3 activate the process P_1 at the end of their execution, as shown in Fig. 4.10.

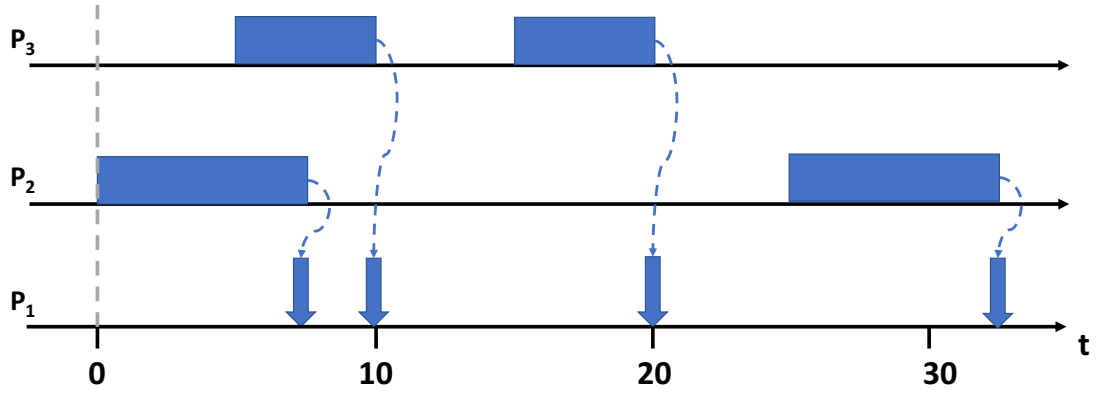


Figure 4.10.: Inter-process Stimulus. Process P_1 is activated at the end of the execution of the processes P_2 and P_3 via an inter-process activation.

4.2.3.2. Single Stimulus

The simplest temporal pattern in the developed model is that of Single Stimulus, which is visualised in Fig. 4.11. It defines a trigger point t_0 at a specific point in time:

$$t_0 = offset.$$

This pattern is intended to describe non-recurring responses such as an initialisation, which is only performed once, e.g., at the beginning of the system's runtime.

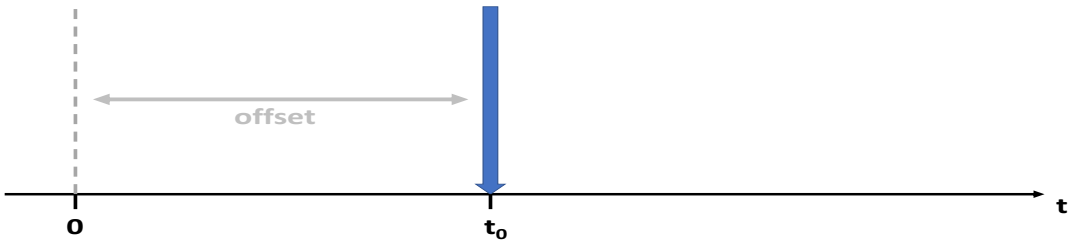


Figure 4.11.: Single Stimulus. A process is activated once after a given offset has passed since the system start.

4.2.3.3. Periodic Stimulus

The most common activation pattern which can be found in real-time systems is that of Periodic Stimulus. It defines trigger points t_i which repeat every *period* time units after an initial *offset*, as shown in Fig. 4.12. Thereby, the period can deviate from its value by a jitter according to a statistical distribution such as a Gaussian Distribu-

tion:

$$t_i = \mathit{offset} + i \cdot \mathit{period} + \mathit{jitter} \quad \forall i \in \mathbb{N}_0.$$

The Periodic Stimulus is used in real-time systems for repetitious activities such as periodically looking for new sensor values or control loops.

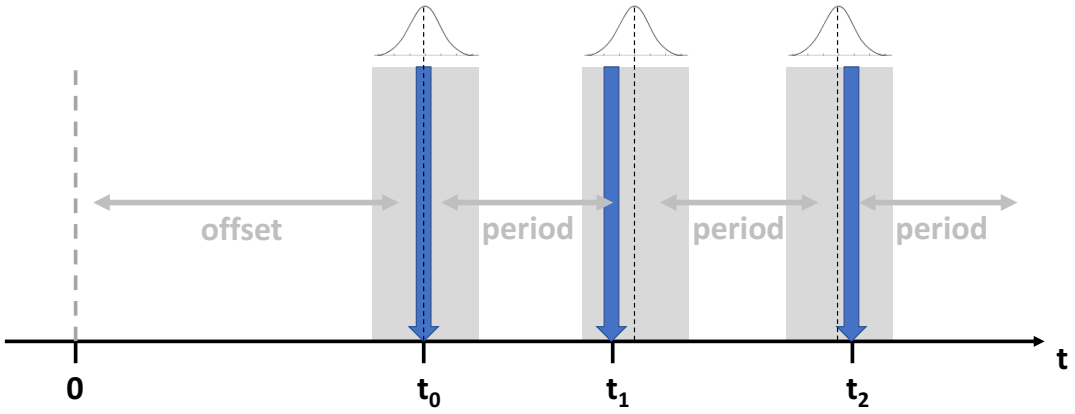


Figure 4.12.: Periodic Stimulus. A process is activated after an initial offset and occurs after that roughly every period of time units. The shown normal distribution depicts the probability that a jitter influences the period.

4.2.3.4. Sporadic Stimulus

Another pattern that also describes recurring activations is Sporadic Stimulus. There, the activation does not happen exactly periodically but deviates slightly according a statistical distribution such as a Gaussian Distribution. This means that each trigger point t_i follows the previous trigger point t_{i-1} after *deviation* time units:

$$t_i = t_{i-1} + \mathit{deviation} \quad \forall i \in \mathbb{N}_0 \text{ and } t_0 = \mathit{offset}.$$

In contrast to Periodic Stimulus, Sporadic Stimulus allows one to express activations that drift away from periodicity. This behaviour occurs, for example, in the clock distribution network of a CPU where a clock does not run at exactly the same rate as a reference clock.

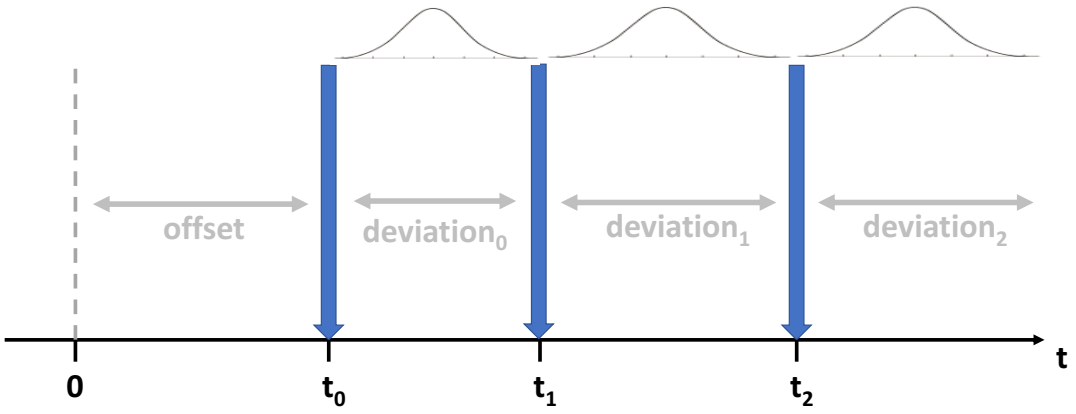


Figure 4.13.: Sporadic Stimulus. A process is activated after an initial offset and occurs after that according the values taken from the depicted normal distribution.

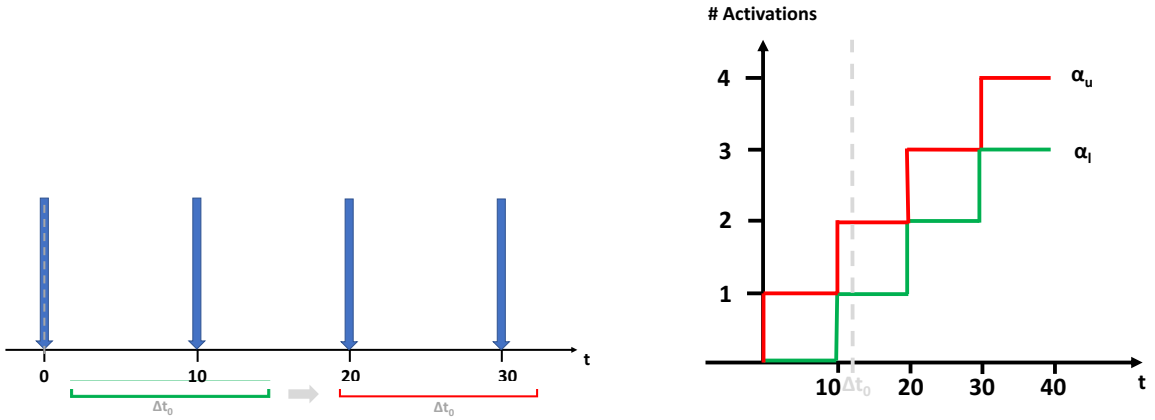
4.2.3.5. Arrival Curves

A more abstract way of defining a pattern is that of Arrival Curves. The concept is based on an event model and allows one to denote ''the total amount of computation that has been requested up to time t" [6, p.1]. In other words, the so-called upper $\alpha_u(t)$ and lower $\alpha_l(t)$ arrival curves define lower and upper bounds for the number of process activations which can be observed in any interval of length t. Hence, Arrival Curves describe the density of activations.

Let A(t) describes the number of activations within $[0;t]$, then the Arrival Curves are defined as

$$\alpha_l(s-t) \quad \leq \quad A(s)-A(t) \quad \leq \quad \alpha_u(s-t) \qquad \forall\ 0 \leq s \leq t.$$

Fig. 4.14a depicts periodic stimulations with a recurrence of ten time units and no offset. If you move a time window $\Delta\, t_0$ of size eleven time units along this sequence of activations, you can always observe at least one activation, but never more then two in this window. Choosing a smaller time frame, however, can lead to sections where no activation can be observed. So, if you continue varying the length of this interval, you get the minimum and maximum number of activations that can be observed at any time. Those bounds constitute the upper and lower Arrival Curves for this periodic stimulus as shown in Fig. 4.14b.



(a) Sequence of activations **(b)** Arrival Curves

Figure 4.14.: Arrival Curve. Example showing the representation of periodic activations with a recurrence of ten time units and no offset as Arrival Curves. The coloured time windows of size t_0 visualise the determination of the lower and upper arrival curve.

4.3. Schedulers

The OS in the developed model is an abstract description of the system software that manages the interaction of computer hardware and software and solely consists of the scheduler.

Definition 4.9 (Scheduling Model)**.** The scheduler model defines a set of schedulers ξ_z, which assign jobs of a task set τ_x to the processing units $\{P_y\}$ of a micro-controller during runtime according a predefined algorithm.

Both the assignment of tasks to a scheduler and the assignment of resources to a scheduler are done before runtime and do not change during runtime. Depending on the mapping of schedulers to the resources, one can distinguish between local, global, and clustered scheduling in multi-core systems ($|\{P_y\}| > 1$). Local scheduling means that each processing units of a micro-controller is managed by its own scheduler ($|\{\xi_z\}| = |\{P_y\}|$). In contrast to this, a scheduling is called global, if a single scheduler ($|\{\xi_z\}| = 1$) manages all processing units of a micro-controller. If a scheduler manages more than one resource then it is possible for a task that is mapped to this scheduler to start execution on different cores with each activation or resume on a different core after its pre-emption. Finally, there is also clustered scheduling, which represents a combination of both. There, schedulers are allowed to manage multiple processing units but the number of schedulers in total is larger than two

($|\{\xi_z\}| < |\{P_y\}| \wedge |\{\xi_z\}| > 1$).

All task sets τ_x and sets of processing units $\{P_z\}$ must be disjunct in the scheduling model. This means that each task and each processing unit is only managed by a single scheduler. As a consequence, hierarchical scheduling is not considered in this work.

A well known scheduling policy in real-time systems is that of task-fixed priority scheduling. There, each job $T_{i,j}$ of a task T_i has the same priority. The prioritisation is done statically before runtime and does not change during runtime. An example algorithm to achieve an initial prioritisation is Rate Monotonic. There, each task is assigned a priority with respect to the inverse of its period, i.e., the task with the shortest period has the highest priority.

An alternative to task-fixed scheduling is job-fixed prioritisation. This means that the priority can differ between the jobs $T_{i,j}$ of a task T_i but it is constant for one job. The best known algorithm for this prioritisation is Earliest Deadline First (EDF). In EDF the priority of a job is inversely proportional to its absolute deadline, i.e., the task with the earliest absolute deadline has the highest priority.

Task-fixed priority scheduling is the most common scheduling policy which can be found in automotive real-time systems. This is not only because task-fixed priority scheduling is the only scheduling policy supported by the AUTOSAR standard so far but also because it provides a low runtime complexity and enables formal schedulability analysis techniques. For these reasons, the focus of this work is on task-fixed priority scheduling.

Besides the prioritisation of tasks, resp., jobs, a scheduling algorithm can also be distinguished by its pre-emptive behaviour. In full pre-emptive scheduling, the active execution of a task can be pre-empted by a task with a higher priority at any point in time. Each pre-emption of a task results in an overhead and increases task execution times because of additional operations that have to be performed such as context switches for saving and restoring a task's data.

Instead, in non pre-emptive scheduling a task executes until completion once it is started. This has no impact on task execution times which makes execution times more predictable but results in an increased blocking time for higher priority tasks.

In order to achieve a trade-off between the benefits and disadvantages of full pre-emptive and non pre-emptive scheduling, there is also the option of cooperative scheduling. In cooperative scheduling, a task can be non pre-emptive for a limited time of the task execution time. This is realised, e.g., by so-called schedule points.

Fixed pre-emption points are defined within a task and pre-emption is limited to these predefined positions. In that way, a task is divided into a set of fixed non pre-emptive sections.

The scheduling model describes just a mapping of task sets τ_x to processing units and does not define additional properties. The properties necessary for scheduling such as pre-emptability or priorities are denoted with the tasks in the software model as described in Sec. 4.2.

4.4. Standards in the Automotive Industry

The fact that automotive software systems are not developed by a single company, but distributed within cross-organisational projects, e.g., between car manufacturers (OEMs) and suppliers (Tier-1s), encouraged the automotive industry to adapt model-based development. A large number of models were published over the last years in order to cover the vast variety of information which arises during the development process. To give an overview on the state-of-the-art of model-based development and to establish a connection to the information covered by our model, three system models that are commonly used in the automotive domain are compared in the following: ASAM MDX, AUTOSAR, and AMALTHEA. Tab. 4.1 gives a compact overview on the models defined by each standard before this section continues with a more detailed introduction.

4.4.1. ASAM MDX

A first version of the model data exchange format (MDX)[1] was released by the Association for Standardization of Automation and Measuring Systems (ASAM) in 2006. Its specified model enables the description of SW-components of an ECU, their interfaces, and data elements. The motivation for ASAM MDX was to get rid of company-specific, proprietary formats and to provide a well-documented, public standard.

The standard is mainly designed for data management and documentation. However, due to the fact that ASAM MDX allows to reference external software parts, which makes the definition of sequential dependencies within the overall system possible, it is also used in the automotive industry for the information exchange between

[1] `https://wiki.asam.net/display/STANDARDS/ASAM+MDX`

OEMs and system suppliers. Furthermore, ASAM MDX strongly influenced the development of the Software Component Template in AUTOSAR, which have both similar use cases.

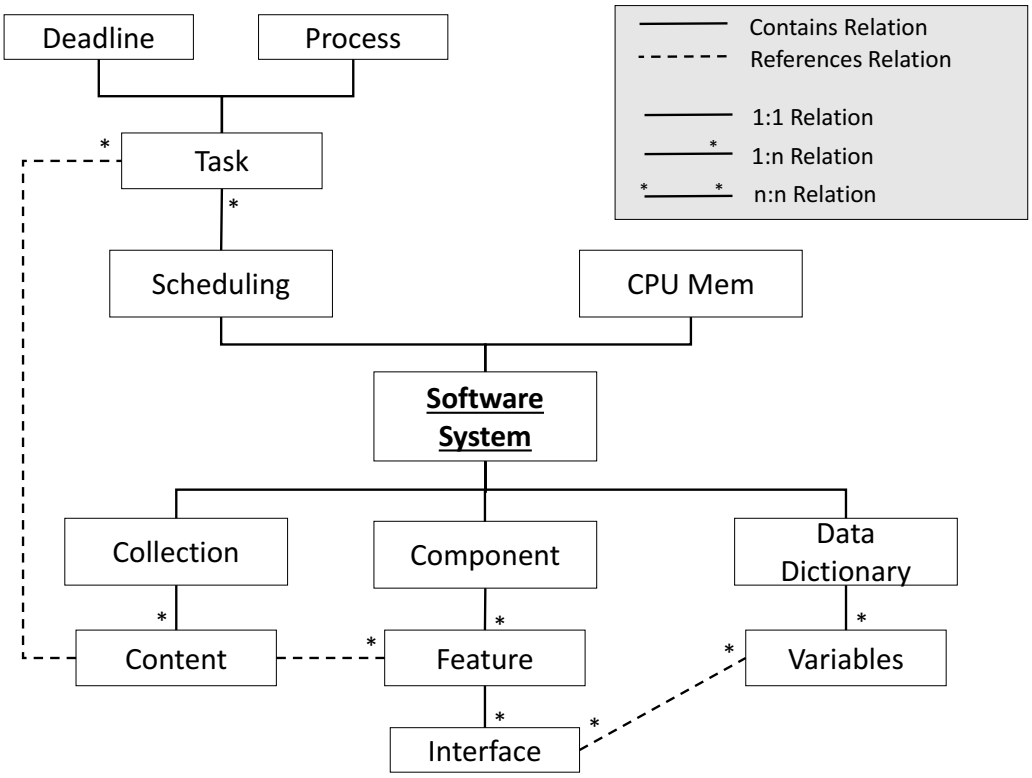


Figure 4.15.: Software System Model of ASAM MDX Entity relationship model visualising the software system meta-model as defined by ASAM MDX. Adapted from [53].

The ASAM MDX standard [53] allows one to define the architecture of a software system for a single ECU. The system under development is described as a hierarchical structure of components. Components in turn encapsulate the internal behaviour provided by functions. Functions, which are called Software Features in ASAM MDX, again contain an abstract model of their implementation. This includes, e.g., an unordered listing of accessed variables and their access type, i.e., buffered or unbuffered, a specification of critical sections for mutual exclusion, and their timed activations. The communication between functions is restricted by interfaces that have to be provided by their containing components.

In addition to this grouping of functionality for application purposes, ASAM MDX also contains execution order constraints and data age constraints to specify dependencies within the software system. The former references external functions which have to be executed before or after a defined indivisible group of functions in a se-

quential manner. In contrast, data age constraints define the maximum age of data consumed by a function, measured from last data production.

ASAM MDX also covers the specification of scheduling units for the operating system. Software functions can be assigned to tasks and either a recommended or a concrete execution order specified. Finally, also details of the software system regarding the required hardware can be modelled, which are basically limited to the memory allocation of the software. Fig. 4.15 gives an overview on the content represented by an ASAM MDX model.

4.4.2. AUTOSAR

AUTOSAR is a standardised software architecture for the automotive industry. One of its main goals is to enable the integration of software from different suppliers in order to increase functional reuse. Therefore, the standard defines a modular software architecture which enables the implementation of software functions independently from the underlying hardware. For example, a developer does not have to bother about the location of a software function when implementing communication, e.g., whether the communication partner resides on the same *ECU* or on an ECU connected via a network. To achieve this in AUTOSAR, software functions communicate with each other via standardised interfaces that are provided by a common software infrastructure, the *runtime environment (RTE)* and the *OS*. This interoperability is possible due to the fact that the common software infrastructure is customised for the system under development by configuring it according to a description of the software architecture descriptions, the so-called *configuration descriptions*. As depicted in Fig. 4.16, the pieces of information required for such a configuration are gathered within the following two models:

System Configuration Description This model provides an abstract description of the overall system or a subset of its components for one ECU. It describes the application software [54] as a collection of *software components* (SW-Cs); components represent architectural elements of the software system which provide interfaces for communication. Each component encapsulates a set of so-called *Runnable Entities*, i.e., sequences of instructions which can be executed and scheduled independently and which establish communication dependencies via the SW-Cs' interfaces. Consequently, a runnable entity captures the actual functionality of the software sys-

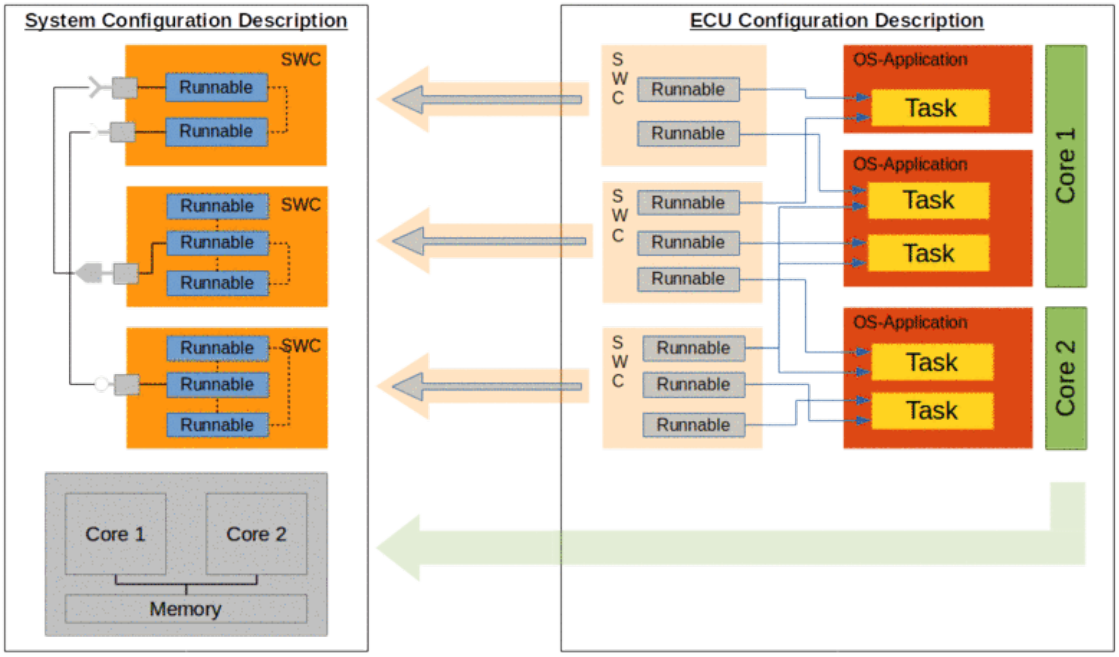


Figure 4.16.: AUTOSAR Configuration Descriptions Schematic visualisation of the information contained within AUTOSAR configuration descriptions. Reprinted from [14].

tem.

The System Configuration Description also contains a model of the actual hardware [55], which hierarchically describes the individual hardware elements of an ECU and their interconnections. For this purpose, AUTOSAR predefines a set of categories for hardware elements, such as *Micro-Controller* or *Processing Unit*, and dedicated attributes, such as memory size.

Finally, the model also contains a set of constraints to define a system's mapping and temporal properties. A mapping constraint describes a mandatory or permitted allocation, e.g., of software functions to processing units or of data signals to memory sections. Timing constraints [56] provide a way to define temporal guarantees or requirements on the system, e.g., the maximum repetition time between the starts of a Runnable Entity or the minimum distance between subsequent data accesses within a given time interval.

ECU Configuration Description The common software infrastructure of AUTOSAR has to be configured in a way that the system under development is managed properly, as described by the *System Configuration Description*. The *ECU Configuration Description* contains concrete values for the configuration of the operating

system and the runtime environment.

The operating system is mainly configured by the definition of tasks, which includes the determination of attributes such as the maximum number of queued activation requests, the priority, and a task's pre-emptability. In addition, the activation patterns for each task are specified and an allocation of tasks to available processing units is performed. The configuration of the runtime environment, which includes, e.g., the mapping of runnable entities to tasks, their positions within in a task, and schedule points, establishes the link between elements of the application software and the operating system.

4.4.3. AMALTHEA

In 2011, the partners of the pan-European research project AMALTHEA[2] started working on a customised, open source tool chain platform. It was motivated by the lack of multi-core support in development tools for embedded systems at that time. The goal of AMALTHEA is to enable an efficient data exchange not only between different cooperating companies, but also between different tools used within a single organisation. The provided tool chain platform including its system model, covers engineering activities such as architectural modelling, partitioning, mapping, and tracing. One of the main focuses of the project, especially its successor AMALTHEA4public, is to make the results open source and thus accessible to everyone in order to establish a de-facto standard. To achieve this, the outcomes of the above mentioned research projects including the tool platform and system model are transferred into an official Eclipse project called APP4MC[3]. Similar to ASAM MDX, the model in AMALTHEA focuses on describing a single control unit instead of a distributed system. Because AMALTHEA is designed to serve as an exchange format, a system can also be described on different levels of detail. The AMALTHEA model is divided into ten sub-models, each covering a specific aspect of the system under development [57]. Fig. 4.17 gives an overview on the covered aspects.

The *Components* sub-model describes software components, which not only encapsulate the internal behaviour provided by functions, but also regulate the communication between each other via ports. In contrast to ASAM MDX and AUTOSAR the description can also contain an unordered listing of the accessed resources like

[2] `http://amalthea-project.org`
[3] `https://projects.eclipse.org/projects/technology.app4mc`

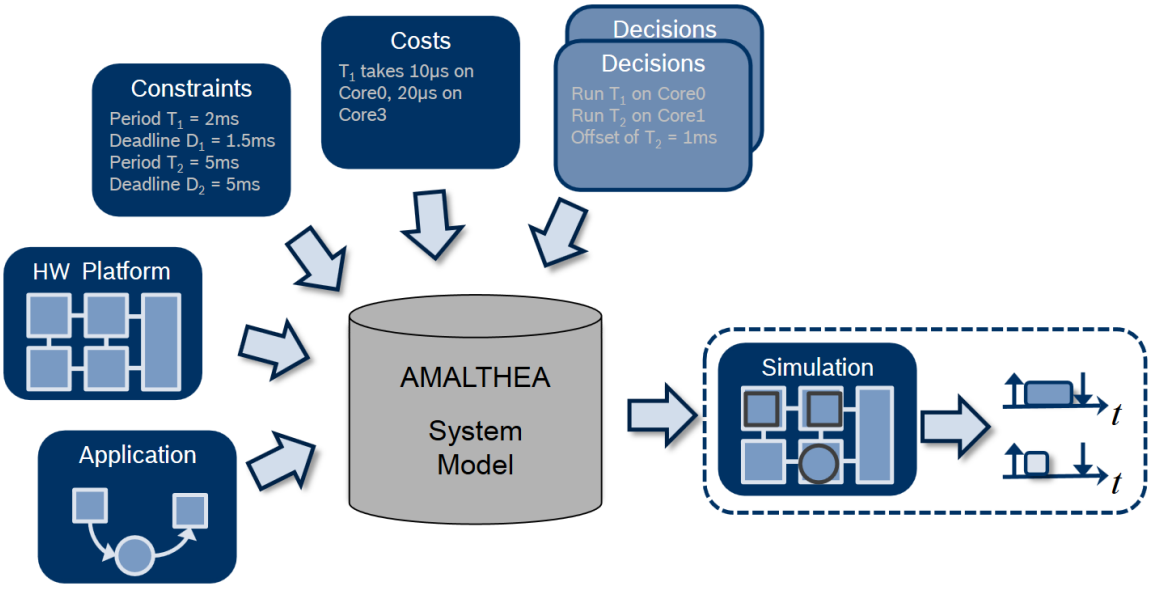


Figure 4.17.: Overview of the contents of the AMALTHEA meta-model. Reprinted from [58].

variables, semaphores, or OS events. In this way, components can be described on different levels of detail, which is important for exchanging information also during early phases of development. Furthermore, a reference to the implementing task can be added for traceability.

The *Software* model describes all parts regarding the functional behaviour of software. This includes details for, e.g., functions, tasks, ISRs, and variables. Although, these pieces of information are basically similar to those within ASAM MDX and AUTOSAR, the level of detail can be higher in AMALTHEA. For example, the software structure in ASAM MDX and AUTOSAR contains just information that variables are accessed. As opposed to this, AMALTHEA allows to state the order and the temporal intervals in which variable accesses occur. Also the control flow can be abstracted by defining the probability of actions such as function calls. This results in a more precise representation of the dynamic software behaviour compared to what is represented in ASAM MDX and AUTOSAR. Still, this representation is not detailed enough to expose the program logic itself.

The *Hardware* model describes the hardware of the system which consists of ECUs, micro-controllers, cores, memories, and additional peripherals. AMALTHEA uses well defined elements instead of generic ones as in AUTOSAR. The *Mapping* model allows to define the mappings of software elements to the operating system and hardware components. In accordance to AUTOSAR, this includes the mapping of tasks

and ISRs to schedulers, which manage the execution on processing cores during runtime. Furthermore, the mapping of data elements such as variables to memory modules is specified.

Event chains and constraints regarding the timing of software, the affinity of functions to each other, or regarding the execution sequence of functions can be expressed with the *Constraints* model. Because these concepts where inherited from TADL, they are in a great part compliant to the Timing Extensions [56] in AUTOSAR. However, the event chain mechanism provided in AMALTHEA works on a hardware related level as discussed later in the description of the *Events* model.

In contrast to the constraints mentioned above, the *PropertyConstraints* model is used to limit the design space for the mapping and allocation decision by providing information about hardware properties required by certain functions. Similar constraints are also available in AUTOSAR. The main difference however, is that in AUTOSAR just constraints for components can be expressed, instead of individual functions.

The *Stimuli* model in AMALTHEA provides a variety of temporal patterns including sporadic, jitter, and arrival curves, which can be used to either activate tasks or ISRs but also to change variable values. In that way it is also possible to model stimulations from the environment, e.g., due to incoming activity on the network bus interface or to represent speed dependent execution in an engine management system. Although this is in a great part also covered by the Timing Extensions in AUTOSAR, AMALTHEA allows to represent the behaviour in more detail using statistical distributions.

With the *Operating System* model, AMALTHEA provides a way to create an abstract description of the OS, like schedulers, scheduling algorithms, resources, and runtime that is produced by an operating system. This is a versatile approach and is not limited to AUTOSAR or OSEK compliant operating systems.

As already mentioned above, the *Events* model is used to provide events that represent specific actions during the runtime of the system. In contrast to AUTOSAR, where events represent an observable change in the system's behaviour on a high level such as different events for each kind of variable access, events in AMALTHEA are considered at a lower, hardware-related level. Therefore, the events in AMALTHEA are specified by the open-source trace format BTF [59]. Then, the events can be used to model event chains, to define timing constraints or to create a trace configuration. The latter can be described with the *Config* model. This model contains definitions

and configurations relevant for simulation or hardware tracing and has no equivalent in AUTOSAR.

Table 4.1.: Models defined by Automotive Standards. Comparison of the models defined by the automotive standards ASAM MDX, AUTOSAR, and AMALTHEA. Reprinted from [14].

	ASAM MDX	**AUTOSAR**	**AMALTHEA**
Considered Version (Release Date)	1.3.0 (June 15, 2015)	4.2.2 (July 31, 2015)	1.1.1 (November 27, 2015)
Scope / Aim	• Meta model for data exchange regarding software module sharing	• Description of a distributed system with multiple ECUs possible • Uniform software architecture • Interfaces for communication • Exchange and configuration formats	• Exchange format for the detailed specification of dynamic software architecture properties • Special focus on multi-core • Extensible and open source tool platform
Hardware Model **• ECUs, cores, memories, peripherals**	**NO**	**YES**	**PARTLY** **• Only one ECU**

	ASAM MDX	AUTOSAR	AMALTHEA
• Data network			• NO
Operating System Model • Abstract OS description • OS resources (e.g. semaphores) • OS runtimes	NO	PARTLY • YES • Only spin-locks • NO	YES
Static Software Architecture • Software component description • Function and data definitions • Communication interfaces	YES	YES	YES
Dynamic Software Architecture • Stimuli (Activation pattern) • Function runtimes • Complex call graph	NO	PARTLY • Only periodic • Only min, max, average • NO	YES
Mapping • Task mapping • Stimuli mapping	NO	YES	YES

	ASAM MDX	AUTOSAR	AMALTHEA
• Data mapping			
Timing Requirements	PARTLY	PARTLY	YES
• Task deadlines	• YES	• YES	
• Generic limits for metrics	• NO	• NO	
• Events & Event Chain requirements	• NO	• YES	
Software Design Constraints	PARTLY	PARTLY	YES
• Execution Order Constraint	• YES	• YES	
• Data Age Constraint	• YES	• YES	
• Data Coherency Groups	• YES	• NO	
• Affinity Constraints	• NO	• NO	

5

Trace Recordings

Analysing the program behaviour of a software system is an essential part in each development process. It is performed repetitiously, e.g., to verify and validate a system's functionality or to measure its performance. For those tasks, developers usually use *debuggers*. These can either be implemented in software or as dedicated hardware. A traditional debugger halts the system's execution and allows one to check the values stored in the memory at that time.

This is possible for cases like the development of desktop software where timing plays a minor part. In real-time systems, however, halting a system's execution might result in a significant alteration of its runtime behaviour. The reason for this is that the correctness of a real-time system is not only dependent on the functional correctness but also on its temporal accuracy. Halting a system's execution using debuggers is consequently inconvenient in the context of real-time systems. Therefore, *tracing* is used in cases where the alteration of timing behaviour is not an option. Tracing, short for "trace recording implies detection and storage of relevant events during runtime[,] for later off-line analysis" [43, p. 1].

5.1. Trace Categories

There are multiple possibilities to detect and store relevant events from the system during runtime. Depending on the realisation, tracing techniques can be categorised into the following three classes, as originally presented in [60]: software, hybrid, and hardware tracing.

5.1.1. Software Tracing

According to [61, 62], software tracing implies the instrumentation of source code in order to gather the desired pieces of information. Instrumentation means that code

Table 5.1.: Trace Technique Categories. Comparison of trace technique categories [19] with respect to cost, mobility, robustness, intrusiveness and safety. The scale runs from very low (– –) to very high (++) accordance.

	Software	Hardware	Hybrid
Cost	++	– –	+
Mobility	++	–	++
Robustness	++	–	+
Intrusiveness	– –	++	–
Safety	– –	++	– –

is added to the source code of the original application which handles not only the detection of relevant situations during runtime, but also their storing or transmission for an eventual analysis. The information can then be transferred to a desktop PC for visualisation and analysis using classical debug hardware or hardware interfaces.

In some cases, software tracing also implies the instrumentation of object code. This is especially common for cases where the source code itself is not available, e.g., in case software functionality is provided by suppliers.

Depending on the way instrumentation is added, the following three situations can be distinguished. In the first one, the required lines of code are added by the developer, so that one speaks of *manual instrumentation*. In case of an *automated instrumentation*, the source code is modified automatically by part of the development tool chain, e.g. by the compiler. Last but not least, there is also *dynamic instrumentation* available. As the name implies, dynamic instrumentation changes the code execution not offline but, in contrast to the two previous ones, during runtime.

In addition to the way instrumentation is added to the system, also the location where it is located can be distinguished. On the one hand, the additional code can be added to the application itself, the so-called *task-level*. This makes it possible to observe events such as function calls and variable accesses. On the other hand, instrumentation maybe placed within the operating system, therefore called *system-level*, so that the interaction between applications, including task switches, and global resources can be monitored.

The big advantage of using software tracing is that it allows an exact control of which events shall be collected. This allows an engineer to tailor the number of observable events to fit a certain use case, e.g., to trace just the communication between

two specific functions instead of all data accesses to that signal.

In contrast to this benefit stands the major disadvantage that instrumentation influences the run-time behaviour of the target application in a negative way. This is due to the fact that additional code has to be executed, which results not only in longer execution times but also in an alteration of event sequences. As a consequence, recorded tracings might not represent the actual behaviour of the system and, thus, be useless. Also the additional utilisation caused by the instrumentation might be critical and lead to deadline violations, if the load of the CPU is already exhausted. Another problem arises in cases, where the software has to be certified according to a safety standard such as [63]. This is due to the fact that each alteration to the application code results in a loss of the certification, which means it has to be reassessed regarding its functional safety.

An industrial product that provides the functionality of software tracing is for example T1 from GLIWA GmbH[1].

5.1.2. Hardware Tracing

In hardware tracing [19, 61], the events are merely generated by hardware, namely by the central processing unit itself. For that reason, dedicated hardware is needed, which supports this kind of tracing. Therefore, this category stands opposite to the approach pursued with software tracing.

Hardware tracing works by closely monitoring the execution of the processing unit at a low level, for which the internal system bus is used. This allows one to observe all necessary information on the internal happenings within a processing unit, such as the executed instructions and the processed data. Corresponding to these two observable pieces of information, two hardware trace approaches can be distinguished: *program flow trace* and *data trace*. A program flow trace, also called *function trace*, stores the execution path of an application during runtime which includes the detection of function calls and taken branches. Instead, a data trace allows one to monitor the state of variables in memory.

The situations where the hardware generates an event, can be configured in advance and depend on what is supported by the processing unit. Finally, the generated events are transferred off-chip. For this purpose, the processing unit needs to have a high-speed interface that is connected to a host computer using another dedicated

[1] `https://www.gliwa.com`

hardware device, the hardware *debugger*. This device not only transfers but also decodes the generated event stream, so that the gathered information can be utilised by a debugger or other third party applications for visualisation and analysis.

In contrast to software tracing, hardware-based tracing bears multiple advantages. It monitors the execution of everything that is happening within the processing unit and not only of the instrumented parts of the software. That way unexpected situations, e.g., memory accesses due to corrupted pointers, are recorded. However, the biggest advantage of hardware tracing is that it works non-intrusively. This means, that no modifications to the actual system have to be made and consequently, no influences to its timing behaviour occur. This is especially important in cases where the system has to be operated in a safety critical environment.

Hardware tracing has also some disadvantages. For one, dedicated hardware is required. Not only does the processing unit have to contain certain features, also additional hardware to transfer the gathered information from the target to a host computer is necessary. Those prerequisites, however, make the use of tracing expensive. For that reason, chip manufacturers offer their chips in two different versions. The first one supports tracing and is designed to be used during development. The other version lacks tracing capabilities and is consequently cheaper, so that it is used for the final product where costs have to be kept low. Furthermore, since hardware tracing monitors the system's behaviour at a very low level and since an exact control of which events are collected is not necessarily possible, the hardware requires extremely huge bandwidths to handle the arising event stream. As a consequence, recording an execution trace that covers the system behaviour comprehensively is very challenging if not impossible due to bandwidth limitations.

In the industry, hardware tracing solutions, meaning the dedicated devices needed to transfer the gathered information from the target to a host computer, are for example the iC6000 by iSYSTEM[2], the PowerTrace-II by Lauterbach[3], or the Universal Debug Engine by PLS[4].

5.1.3. Hybrid Tracing

Hybrid tracing [62] finally bridges the gap between software and hardware tracing. As the name already implies, there the tracing is realised as a combination of both soft-

[2] `http://isystem.com`
[3] `http://www.lauterbach.com`
[4] `https://www.pls-mc.com/`

ware and hardware. This technique is based on the fact that some processors allow one to re-program their microcode. The microcode is then re-programmed in such a way that tracing is enabled for the instructions relevant to trace. As a consequence, once this microcode is executed, the tracing is triggered and an unique value representing this instrumentation point written to a specific memory location or I/O port. A hardware device connected to the processor collects these instrumentation point identifiers, timestamps and, finally, records them in a trace.

Hybrid tracing combines the best of the tracing techniques mentioned previously and succeeds in minimizing the instrumentation overhead. Nevertheless, it also inherits one of software tracing's crucial disadvantages, namely that it works intrusively. Due to this fact that the runtime behaviour of an application is modified, this technique is also not tolerable, if the software is developed with respect to certain safety standards.

An industrial product, which provides hybrid tracing is, e.g., RTBx by Rapita Systems[5].

5.2. Trace Techniques

While different categories of tracing were introduced in the previous sections, this section presents in detail the individual ways how tracing data can be obtained. Note, that the following techniques [64] do not necessarily have to be considered individually, but they can also be realised as a mixture of multiple ones.

Table 5.2 compares the following trace techniques with respect to required trace hardware, feasible size of trace recordings, intrusiveness, and impact on timing. In summary, it can be stated that engineers have to make a compromise between the amount of information that has to be stored in a trace and the impact on the system's timing when choosing a trace technique.

According the experience of Timing-Architects, software target trace and on-chip trace are currently the most frequently used techniques in industrial projects. The former is mainly applied in situations where a particular information is required, such as a data access by a specific function. On-chip trace, instead, is used to get a continuous insight in the system's behaviour, e.g., the interaction of tasks.

[5] `https://www.rapitasystems.com`

Table 5.2.: Trace Techniques. Comparison of trace techniques with respect to required trace hardware, location and size of trace recordings, intrusiveness, and its impact on timing as presented in [64, p. 9]. The scale runs from very low (– –) to very high (++) accordance or, respectively, yes (✓) and no (×).

	Trace HW	Trace Size	Intrusive	Timing Impact
Bus Trace	✓	– ...+	✓ / ×	– – ...++
Flow Trace	✓	– –	×	– – ...–
On-Chip Trace	×	– –	×	– –
Extended On-Chip Trace	×	–	×	– ...++
Software Trace Target	×	– ...++	✓	–
Software Trace Host	×	++	✓	–
Snooper Trace	×	– –	✓	–
Advanced Register Trace	×	– –	×	– –

5.2.1. Bus Trace

Bus trace [64, p. 1 ff.] describes a technique, where information transferred via the external system bus of the target is recorded. Due to the fact that the system bus consists of an address, data and a control bus, all necessary pieces of information to deduce the internal behaviour of a system can be observed. This includes the program flow, all data accesses and their values.

The bus trace technique is mainly used for micro-controllers without internal periphery and memory. However, it requires physical access to the memory interface in order to be able to observe the necessary signals. This can either be implemented in the CPU or in the memory module. A problem arises in cases where the CPU has a cache or internal RAM, in addition to the external memory module. Then, not all ongoing internal events can be observed due to the fact that data is stored in the internal memory for faster access. Instead, only the data exchange between the cache and the external memory can be recorded. This deficit can, e.g., be resolved by either completely turning off the cache, which then results in lower performance, activating write-through for the cache such that all changes to the data are replicated in the

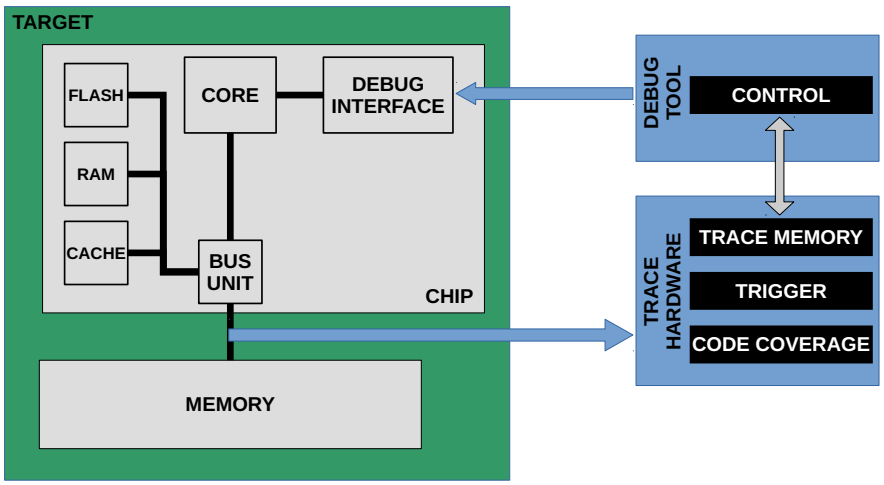


Figure 5.1.: Bus Trace. Adapted from [64, p. 3].

external memory, or by code instrumentation to redirect data accesses to the cache.

In order to observe the system bus during runtime comprehensively, a large number of signals have to be recorded simultaneously. For that reason, a bus trace is in general realised by a *logic analyser*, which is an electronic instrument that provides not only a great number of inputs but also the possibility for high performance data recording.

5.2.2. Flow Trace

The trend in the chip industry to put all peripheral parts including the memory together with the processing units on a single chip (System-On-Chip) has resulted in the fact that external interfaces to the memory module are not available any more. Consequently, performing a bus trace is infeasible. Instead, recent CPU architectures provide a dedicated *trace interface* in addition to a debug interface [64, p. 3 ff.] which can be seen in Fig. 5.2.

This interface is used to make the information transferred via address and data bus available to the outside in a compressed way. Thereby, the complete program flow and all data accesses including their values can be observed. Current interfaces use a four to 16 bit wide trace bus which is operated at a maximum frequency of around 400 MHz. Some trace protocols also allow one to encode the transferred bit stream in real-time, which make analyses for triggering and code coverage possible.

The biggest disadvantage of the flow trace is that it has to struggle with the avail-

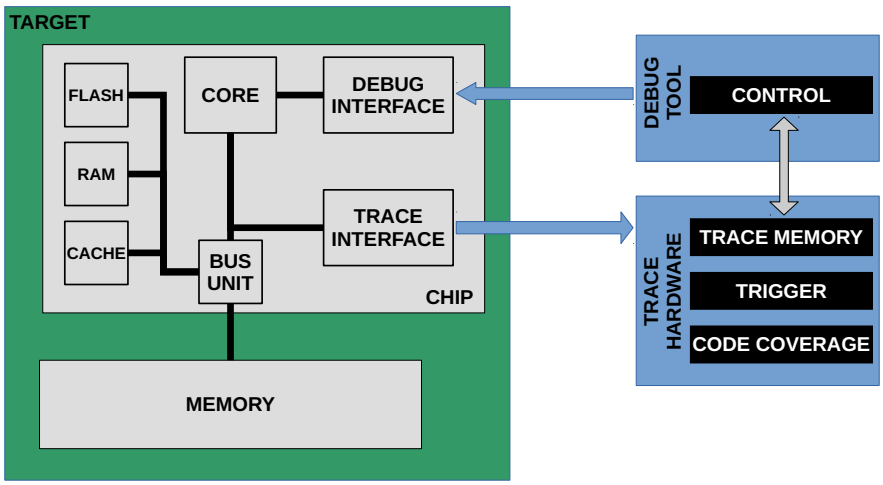


Figure 5.2.: Flow Trace. Adapted from [64, p. 5].

able bandwidth, which is very limited compared to the amount of data. While recording the complete program flow can be done without any problems, trace interfaces quickly reach their limits if also data accesses have to be included in the trace. In order to tackle this problem, some chips have an additional FIFO buffer or just specific program parts or variables are traced. For that reason, some mainly contemporary interfaces provide also a way to define address ranges, which then are included or, respectively, excluded from the trace.

5.2.3. On-Chip Trace

For *on-chip tracing* [64, p. 5] some CPUs contain a trace memory instead of a dedicated trace interface as depicted in Fig. 5.3. This trace memory is implemented as an additional FIFO buffer, in which the last addresses of executed jump instructions and their destinations are recorded. The gathered information can be transferred off-chip via a debugging device, where it can be used to deduce the program flow. As a consequence, a dedicated trace interface can be omitted.

The biggest disadvantage of this technique is that only a limited amount of memory is available to collect the trace information. This is due to the fact that the price as well as the used die space of the chips have to be kept small. As a consequence, currently a maximum of around 1.000 instructions can be stored only.

To eliminate this major shortcoming of an on-chip trace, *extended on-chip trace* [64, p. 6] can also take advantage of the available RAM. In case of a full trace buffer, an

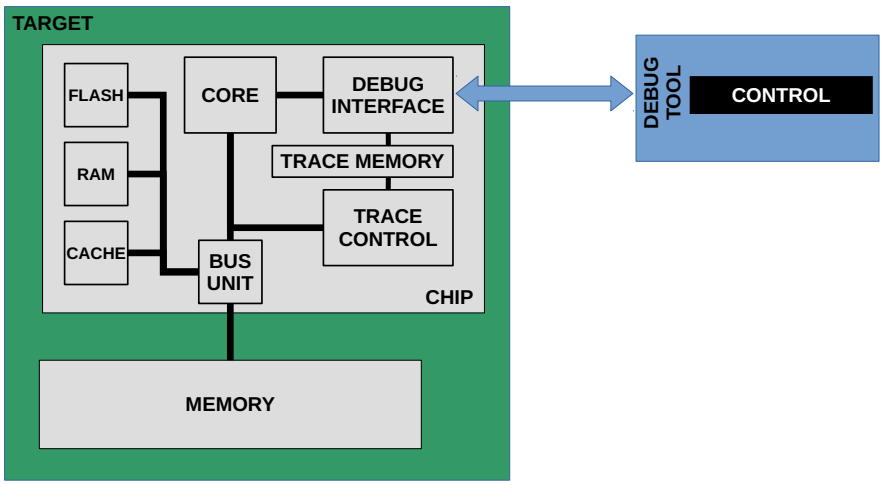


Figure 5.3.: On-Chip Trace. Adapted from [64, p. 7].

interrupt service routine is called, which then copies the data into the RAM. After that, regular program execution continues. Although this technique makes it possible to store an additional amount of trace information, the executed interrupt service routines have a considerable impact on the run-time behaviour of the system. Alternatively, the trace information can also be written directly to the RAM instead of copying it from the trace buffer. For this purpose, some CPUs support direct memory access so that the influence on the actual run-time behaviour of the system is kept to a minimal.

5.2.4. Software Trace Target

Instead of depending on the hardware to support gathering information about the runtime behaviour of the system, the trace memory, a dedicated area in the target's RAM, can be filled by the application software itself [64, p. 6]. For that reason, the developer adds functions to the application which store the trace information in the memory in a well defined structure. Finally, the gathered information can again be transferred and analysed off-chip via a debugging device.

The advantage of this technique is that the developer has full control on what pieces of information shall be gathered. This includes not only individual addresses and their values but, more importantly, also operating system constructs like processes or complex data structures. That way a tracing solution tailored to the current situation can be developed.

5.2.5. Software Trace Host

The *software trace host* technique[64, p. 6 f.] is basically the same as the previous one above. However, the gathered information about the runtime behaviour of the system is not written into a dedicated area in the target's RAM but into the memory on a connected host system. TO do so, the application has to be instrumented in such a way that the arising information is sent to the host via one of the available communication interfaces using a specific protocol.

The debug interface is predestined for this communication, since it is connected to the target anyway. Trace data is then written to a dedicated buffer, which is either directly accessed by the debugger without any timing interference, if this is supported by the CPU, or periodically while the application is stalled. For the latter, a ring buffer structure can be used to minimise its impact on timing.

5.2.6. Snooper Trace

Snooper trace [64, p. 8] is a simple technique where the memory is periodically checked for changes during program execution. Modifications of data values and the program counter can be observed and the runtime behaviour of the system deduced accordingly. This technique is especially convenient for CPUs that provide the debugger direct access to the target memory without halting system execution. Even then, the quality of the resulting trace is highly dependent on the frequency, which is not only given by the used debug interface but also the amount of observed data. Hence, this technique is most suitable for systems where values do not change too rapidly; otherwise, it results in gaps and data loss.

5.2.7. Advanced Register Trace

For an *advanced register trace* [64, p. 8] the current content of the CPU registers is logged on the host system at every "STEP", "GO" or "BREAK" instruction at assembler level. Based on these pieces of information, the complete program flow can be deduced. By applying this technique, currently, up to 65.536 steps can be recorded.

5.3. Trace Format

A major challenge, which all existing trace techniques have in common, is that engineers have to find a reasonable compromise between the amount of information that has to be stored in a trace and the impact on the system's timing. As a consequence, two approaches evolved in the industry to meet this challenge. In the first approach, which is called *program flow trace*, the system is considered at *process level*, i.e., the interaction of processes is monitored. Because the resulting amount of information is reasonable, this is done via hardware or software trace with neglectable impact. Depending on the frequency of process occurrences, state-of-the-art techniques allow one to record in addition all function calls. The other approach is called *data trace* and stores information at *system level*, where operating system specifics such as data and semaphore accesses are monitored.

To go into more detail on which actions can be monitored on process level and system level, an introduction to a trace format is given in the following. We have chosen BTF as trace format. The main motivation for our choice is that it is designed for timing evaluation of event based systems, which has a beneficial effect on the reverse engineering. Furthermore, the BTF specification [59] is publicly available and, thus, can be employed by anybody to reproduce our results.

5.3.1. Process Level

The temporal behaviour of a real-time system is primarily determined by the employed scheduling policy, by which the operating system decides which process can run on which processing unit. Processes can evolve through the following states during a system's execution [59]:

- ACTIVE: The process is ready to be executed and waits for allocation of a processing unit for the first time.
- NOT INITIALIZED: The process is passive and can be activated.
- PARKING: The process has requested a resource that is unavailable and has been preempted while waiting for the resource's release.
- POLLING: The process has requested a resource that is unavailable and waits actively for the resource's release.
- READY: The process fulfils all functional prerequisites for execution and waits to be allocated a processing unit.

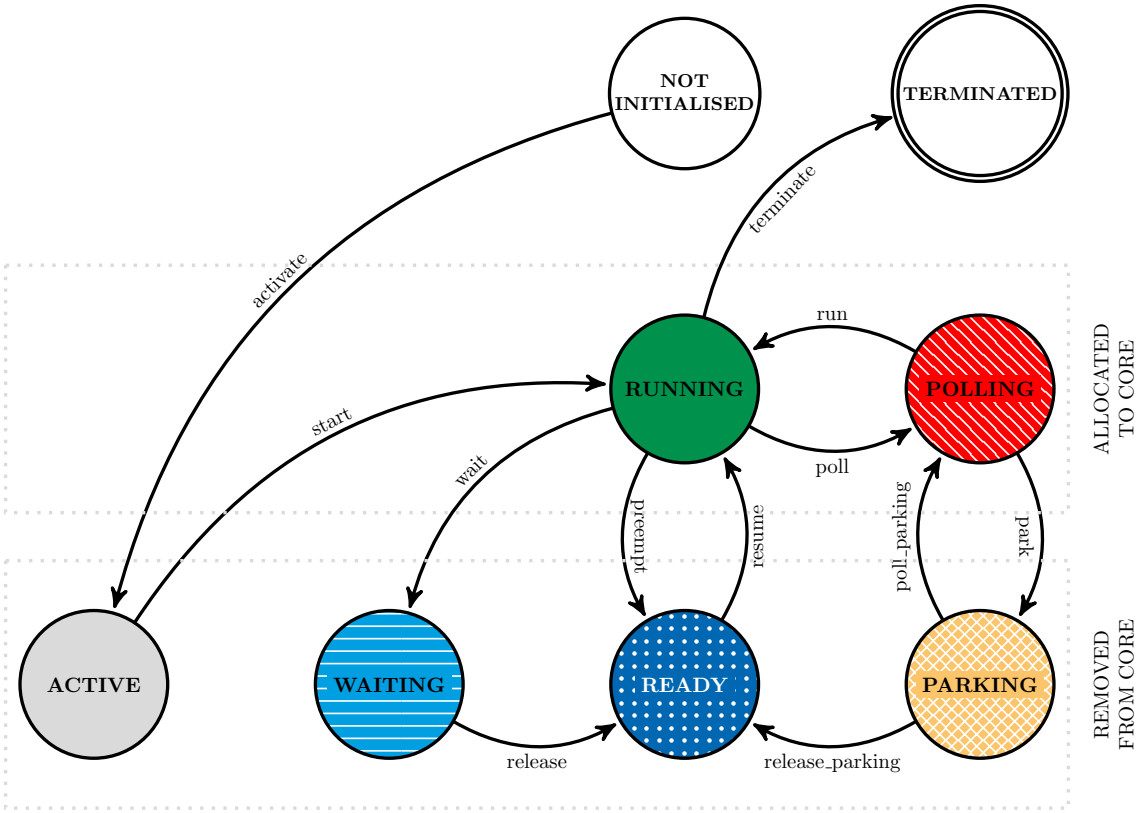


Figure 5.4.: Process State Model. AUTOSAR task state model, extended by states for the active (POLLING) and passive (PARKING) waiting for a Resource. Adapted from [59, p. 13].

- RUNNING: The process has been assigned a processing unit, and its instructions are being executed.
- TERMINATED: The process finished executing its instructions and is passive.
- WAITING: The process has requested an OS Event that is unavailable and waits passively for the event's occurrence.

These states and their transitions to each other are depicted in Fig. 5.4. In this figure, the basic task state model employed by AUTOSAR [65], which is adopted from the OSEK standard [66], is refined in order to distinguish some task states more precisely. Originally, the OSEK basic task state model captures only the following three states [66, p. 18]: running, ready, suspended, and waiting. This means, that the states *NOT INITIALIZED* and *TERMINATED* in Fig. 5.4 are equal to the OSEK state *suspended*, the states *ACTIVE*, *PARKING*, and *READY* correspond to the OSEK state *ready*, and the state *RUNNING* together with the state *POLLING* capture the OSEK state *running*.

Because processes are implemented in the operating system as function calls, events on process level are monitored, in general, via program flow trace. A program flow trace allows to mark function addresses for observation and if the program calls or returns from one of the marked addresses during execution, a timestamp is recorded. This allows one to reproduce the program flow and to analyse the interaction of processes. Thanks to the capabilities of the latest trace hardware and interfaces, it is also possible to record the function calls performed by each process, i.e., the execution of runnables, if the amount of arising information is not too large. Runnables are executed within the body of a process and traverse similar states:

- NOT INITIALISED: The runnable is passive and can be started.
- RUNNING: The runnable executes on a processing core.
- SUSPENDED: The runnable stopped execution on a core.
- TERMINATED: The runnable finished executing its instructions and is passive.

Fig. 5.5 depicts the state model of a runnable according to BTF [59]. It represents the states defined by AUTOSAR [67, p. 126] in a condensed way by not distinguishing

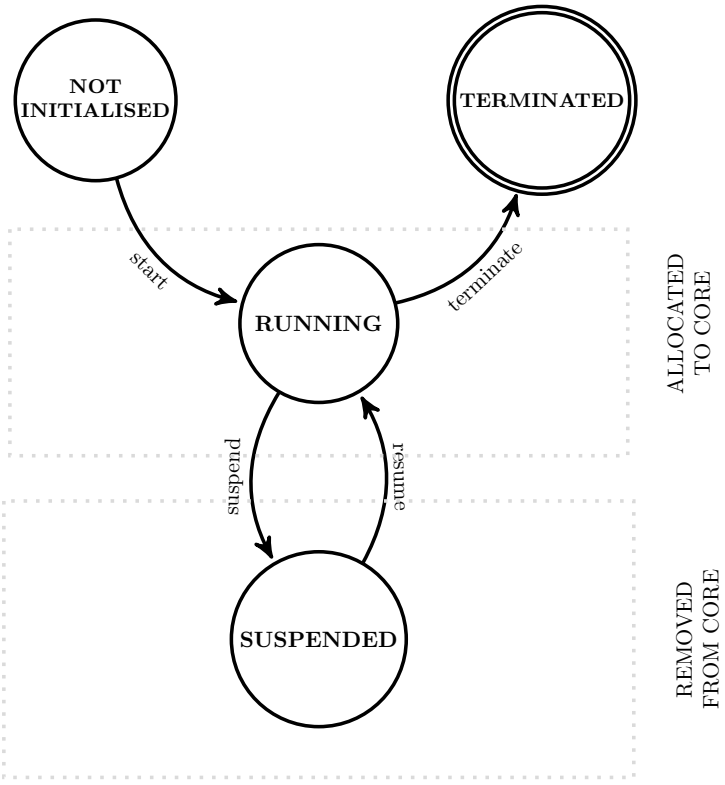


Figure 5.5.: Runnable State Model. Adapted from [59, p. 17].

between the states *waiting* and *preempted*. This is due to the fact that runnables are executed within the context of a process and, thus, the missing states from AUTOSAR can be inherited from the state of the calling process. All other states are equally available in AUTOSAR. Thus, the states *preempted* and *waiting* in AUTOSAR correspond to the state *SUSPENDED* in BTF, the state *to be started* to the state *NOT INITIALISED*, and the state *suspended* to the state *TERMINATED*.

5.3.2. System Level

To perform a program flow trace, it is enough for the trace logic to monitor the start address of functions, which is a manageable amount. But to get more information on how the processes are interacting with the system and with each other, all accesses to the data memory have to be observed. Moreover, the trace logic must not only register accesses to specific memory locations but also the modifications performed, i.e., the data values set. Because a program flow trace cannot provide this, the so-called data trace must be employed in order to get a trace on system level.

Modifications to the memory are performed within the context of a process. This means all actions happen in the running state of a process and, thus, represent transitions that result in the same state again. The following actions on system level are defined by BTF and can be detected via data trace:

Signal Actions A signal represents an address in the memory of a microcontroller. This memory location contains a value, which can be accessed in two different ways by a process:

- read: The data value stored at the defined memory address of the signal is read by a process.
- write: A data value is written to the signal by a process, i.e., the memory address defined by the signal is set to the stated value.

OS Event Actions OS events are a mechanism for synchronisation provided by the operating system. The following three actions can be performed by a process to achieve synchronisation:

- clear_event: The stated OS event is cleared so that processes have to wait again for the occurrence of this OS event.

- set_event: The stated OS event is set by a process in order to notify waiting processes that a defined progress in execution is reached.
- wait_event: A process waits at a predefined position during execution for the occurrence of the stated OS event, e.g., if this process requires information that is provided by another process.

Semaphore Actions A semaphore is a mechanism provided by the operating system for limiting the number of concurrent accesses to a resource. The following two actions can be performed by a process to synchronise communication:

- request: A process requests the stated semaphore and the number of remaining concurrent accesses is decremented.
- release: A process releases the stated semaphore and the number of remaining

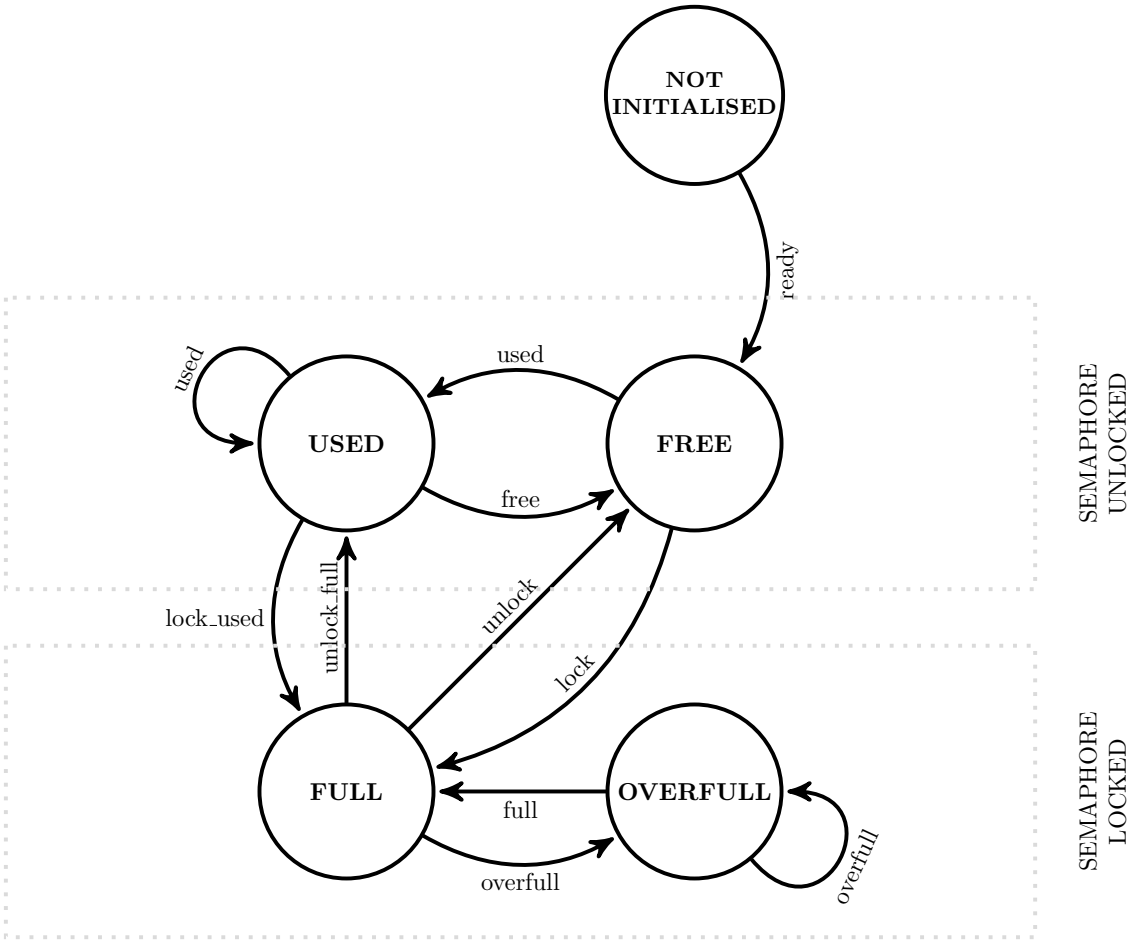


Figure 5.6.: Semaphore State Model. State model of a mutex, resp., counting semaphore for handling concurrent accesses to resources. Adapted from [59, p. 22].

concurrent accesses is incremented.

A semaphore is defined by its initial value n, which states the maximum number of concurrent accesses. If this number is exceeded, the access to the stated resource is blocked and the requesting process has to wait until one of the previous accessing processes releases the resource. This means, that a semaphore can also change its internal state. The following states of a semaphore are defined:

- FREE: No process has requested the semaphore.
- FULL: The semaphore has reached the maximum number of requests.
- OVERFULL: The semaphore has reached the maximum number of requests and at least one process is waiting for the semaphore.
- USED: Processes have requested the semaphore but its maximum number of concurrent accesses is not reached.

Fig. 5.6 depicts the state model of a mutex, resp., counting semaphore according to BTF [59].

5.3.3. Trace Events

The BTF standard defines not only the actions that can be monitored during a system's execution but also a representation of the trace for later off-line analysis. Besides storing the information in a database to achieve a high performance during analysis, the standard also supports listing a trace in a textual and readable manner. Def. 5.1 gives an explicit structure that allows one to make an unambiguous statement about the system's internal behaviour based on so-called events.

Definition 5.1 (Event)**.** Let <*timestamp*> be a time measure, such that a chronological order of events can be achieved, <*source*> and <*entity*> the identifiers of an element in the system, <*sourceInstance*> and <*entityInstance*> an identifier, that makes it possible to distinguish coexisting elements, *type* an identifier, that represents the type of the element stated by <*entity*>, and let *action* be an identifier, that indicates what happened in the system, then an event <*event*> is defined by the septuple

<event> := (<timestamp>,
<source>, <sourceInstance>,
<type>,
<entity>, <entityInstance>,
<action>).

An event is defined by seven attributes, which are all separated by commas (CSV). The first attribute <*timestamp*> indicates the moment in time when the event occurred. It represents the amount of time that elapsed since the beginning of a trace recording. This continuous counter in a used-defined time unit, which is set to nanoseconds by default, makes it possible to achieve a chronological order of the system's internal behaviour.

The attribute <*action*> denotes a change to the system that was observed during execution. As discussed before, each action corresponds to a state transition of an element that is monitored on process or system level.

The remaining attributes *source*, *sourceInstance*, *type*, *entity*, *entityInstance* identify the element that changed and, thus, that caused the event. Both *source* and *entity* represent identifiers of an element in the system and reference the according process, function, semaphore, or data signal by its name. The source describes the context of the affected element, e.g., the task in which a runnables is started. This is essential for elements that are accessed from multiple contexts, which is usually the case. The attributes <*entityInstance*> and <*sourceInstance*> are integers that allow one to distinguish coexisting instances of the same element. This is of importance for elements with a life cycle, such as processes or runnables, which can be executed in parallel. Because the names of actions are not ambiguous, e.g., a start action exists for processes and runnables, the attribute *type* denotes an abbreviation (I, R, SEM, SIG, T) that corresponds to the type of the affected element in the system (ISR, runnable, semaphore, signal, task).

A trace recording is, consequently, defined as a sequence of trace events, as it is shown in the following example.

Example Trace (FMTV Challenge 2016): Listing 5.1 contains an extract of a simulation trace produced by the TA Simulator in order to show how a BTF trace of an actual system looks like.

```
1996021,Angle_Sync,0,SIG,Label_7442,0,read,0
```

```
1996021,CORE1,0,T,Angle_Sync,0,poll
1996106,CORE1,0,T,Angle_Sync,0,run
1996106,Angle_Sync,0,SIG,Label_8386,0,read,0
1996106,CORE1,0,T,Angle_Sync,0,poll
1996191,CORE1,0,T,Angle_Sync,0,run
2000000,periodic_1ms,2,T,Task_1ms,2,activate
2000000,Angle_Sync,0,R,Runnable_6660us_48,0,suspend
2000000,CORE1,0,T,Angle_Sync,0,preempt
2000000,CORE1,0,T,Task_1ms,2,start
2000000,periodic_2ms,1,T,Task_2ms,1,activate
2000000,Task_1ms,2,R,Runnable_1ms_0,2,start
2000000,Task_1ms,2,SIG,Label_849,0,read,0
2000000,CORE1,0,T,Task_1ms,2,poll
2000085,CORE1,0,T,Task_1ms,2,run
2000085,Task_1ms,2,SIG,Label_5861,0,read,0
2000085,CORE1,0,T,Task_1ms,2,poll
2000170,CORE1,0,T,Task_1ms,2,run
```

Listing 5.1: Extract of a Simulation Trace produced by the TA Simulator from the AMALTHEA Model provided for the FMTV Challenge 2016

The model used for this timing simulation is the AMALTHEA model provided for the Formal Methods for Timing Verification (FMTV) Challenge 2016 [68]. It describes a full-blown engine management software and is discussed in full detail in Sec. 8.3.1.3. The listing shows the system's internal behaviour at around 2 ms (2000000 ns), where the periodic tasks *Task_1ms* (line 8) and *Task_2ms* (line 12) are activated. The former task is activated for the second time and the latter for the first time, which is denoted by their instance counters. Because *Task_1ms* has a higher priority, the currently running task *Angle_Sync* is preempted in line 10.

5.3.4. Database Representation

Besides the possibility described above to store BTF information as comma-separated values (CSV), it is also possible to use a database. This allows one not only to search efficiently for individual information such as all activated tasks within a certain time frame, but also to optimise access to the data for recurring queries.

An example for the implementation of a BTF-compliant database design is the

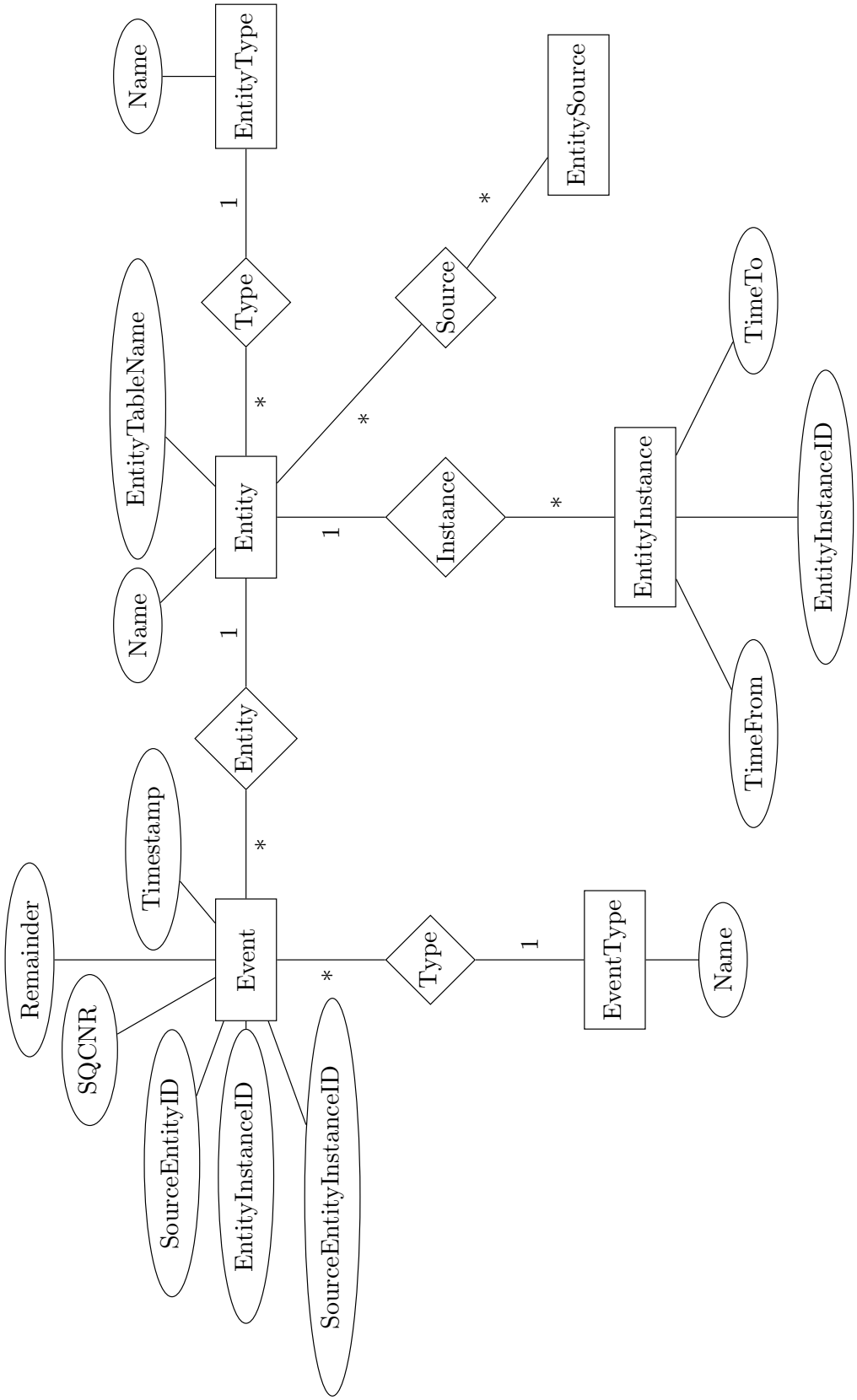


Figure 5.7.: Entity-Relationship Model of ATDB. BTF-compliant Database Design of the APP4MC Trace Database (ATDB).

AMALTHEA trace database (ATDB) [58]. Its design is optimised for the analysis of entity correlations and temporal metrics. As a consequence, parts of the information are stored deliberately in a redundant way in order to enable efficient data requests. The database model of ATDB, which is shown in Fig. 5.7, consists of the following mixture of statically and dynamically defined tables:

- **Entity** contains information about all the entities recorded in the trace including the name of the entity and a reference to its type, which is stored in the EntityType table. Because the Event tables are created dynamically, it also stores the name of the table in which the events for that entity are stored.
- **EntityType** contains all the types of entities stored in the database, e.g., Runnable, Signal, or Task.
- **EntitySource** stores information about which entity causes an event by another entity.
- **EntityInstance** contains information about the lifetime of each instance of an entity by storing the time stamps from and to which it was active. Each entry also has a reference to the affected entity and an instance number that shows how often this entity has been activated up to that point.
- **Event** tables are created dynamically in a variable amount. A table is created for each entity and contains all the events of that entity including the timestamp at which the event occurred and a reference to entity that caused the event. The information which Event table stores information of which entity is part of the Entity table.
- **EventType** contains all the actions, i.e., the types of events performed by the entities stored in the database, e.g., activate, start, or terminate.

5.3.5. Trace Analysis

The main purpose of trace recording is the detection and storage of relevant events during runtime for later off-line analysis. Listing 5.2 shows a possible sequence of observed state transitions for a fictional task $Task_1$.

```
5,Alarm,1,T,Task_1,1,activate
15,Core_1,0,T,Task_1,1,start
20,Core_1,0,T,Task_1,1,poll
25,Core_1,0,T,Task_1,1,park
```

```
30,Core_1,0,T,Task_1,1,release_parking
35,Core_1,0,T,Task_1,1,resume
45,Core_1,0,T,Task_1,1,preempt
55,Core_1,0,T,Task_1,1,resume
65,Core_1,0,T,Task_1,1,terminate
75,Alarm,2,T,Task_1,2,activate
```

Listing 5.2: Simple Trace Example

This trace respects the BTF [59], where each line stands for an observed event. In this example, each line represents a transition of the task $Task_1$ from one state to another according the state machine presented in Fig. 5.4. Together with the timestamp, these pieces of information can be used to visualise the recorded system execution in a Gantt chart, as depicted in Fig. 5.8.

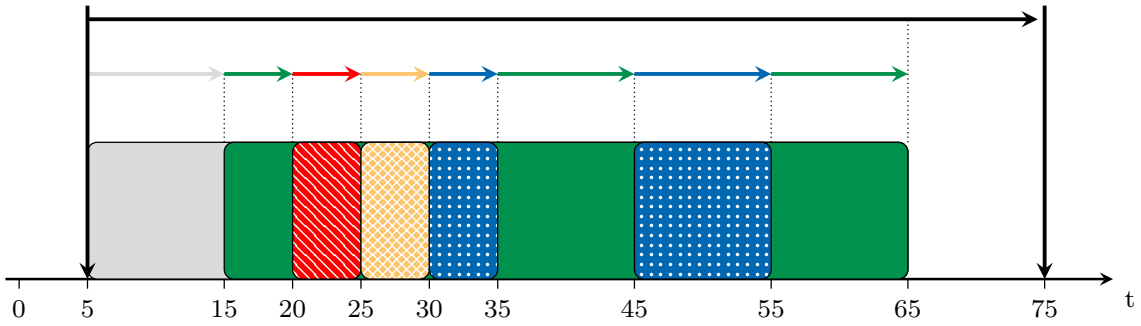


Figure 5.8.: Gantt Chart of Trace Example. Visualisation of the trace example of Listing 5.2 in a Gantt chart, showing task $Task_1$ in different states over time. Adapted from [13].

This figure shows the temporal sequence of the states in which a task resides over time as bars on a time line. To be able to distinguish the individual states in the Gantt chart, each bar is coloured and patterned according the colour, resp., pattern of the corresponding state defined in Fig. 5.4. With the help of such a Gantt chart, it is possible to reproduce the interaction of individual elements in order to understand the internal behaviour of the system. This means in the case of our example, that it allows one to understand why it took task $Task_1$ 60 time units to finish although it executes only 25 time units on the processing core. As a consequence, the engineer can look into which other task requested the shared resource that caused the task to poll and by which task it was preempted later on.

Besides visualisation, another way to objectively analyse a system's temporal be-

haviour is to calculate metrics [69] from a trace recording:

- A2A: *Activation-To-Activation*: The distance between two successive activations of a task. In Fig. 5.8, this is indicated by the horizontal black arrow.
- NET: *Net Execution Time*: The actual execution time of a task, which spans from the task's start to its termination and excludes the time the task is preempted. In Fig. 5.8, this is indicated by the sum of the lengths of all green arrows.
- Parking: *Parking Time*: The timespan of a preempted task waiting for a requested resource. In Fig. 5.8, this is indicated by the length of the orange arrow.
- Polling: *Polling Time*: The timespan of a task actively polling for a requested resource. In Fig. 5.8, this is indicated by the length of the red arrow.
- Ready: *Ready Time*: The timespan of a task between its start and termination, in which it is not executed on any processing unit. In Fig. 5.8, this is indicated by the sum of the lengths of all blue arrows.
- SD: *Start Delay*: The time from the activation moment of the task to the moment of its start. In Fig. 5.8, this is indicated by the length of the grey arrow.
- Waiting: *Waiting Time*: The timespan of a task actively waiting for an OS Event. This metric is not indicated in the Fig. 5.8 because according events are missing in the example trace for simplicity reasons.

The listed metrics allow one to assess a system's real-time performance and that way to comprehend its internal behaviour. They represent only a selection of the variety of metrics that have been defined by the community over the years. Furthermore, this example considers only a trace recording at process level. In general, this is sufficient to comprehend the temporal behaviour of a real-time system, because tasks are the smallest schedulable units managed by the operating system. However, besides tasks also other entities, e.g., functions and data signals, as well as their corresponding events, such as function calls and data accesses, can be observed during a system's runtime. With the help of these pieces of information and according metrics, it is possible to get an even more detailed insight into the system's behaviour.

Part II.

Contributions

6

CoreTAna

CoreTAna is the name of our novel tool that derives an AUTOSAR-compliant model of a real-time system by conducting dynamic analysis using trace recordings. It is part of the TA Tool Suite [70] as an experimental feature and allows one to generate a model at system level. This includes a description of the static information on a system such as the tasks and their mapping to the processing cores but also dynamic information such as the runtime behaviour and data dependencies. A limited version of CoreTAna that synthesises just a task model is included in the open-source Eclipse APP4MC[1] tool chain.

This chapter elaborates on CoreTAna's internal workings as they are implemented in the TA Tool Suite Release 16.03 which is the version used for all evaluations and case studies mentioned in this work. A detailed description of the algorithms employed by our reverse engineering approach are presented next.

6.1. Design

As depicted in Fig. 6.1, CoreTAna's reverse engineering starts with a system's captured hardware traces. Because vendors use their own trace format, the traces are first transformed into the BTF trace format [59], which provides a common interface for CoreTAna (see ❶ in Fig. 6.1). A SQLite database is created for each trace recording and the events are stored in such a way that querying for existing elements or for events of specific entities can be performed quickly.

Inputs to the reverse engineering are also other already available pieces of information of the model, e.g., the program structure derived from static analysis or a detailed description of the known hardware platform. For each entity that is contained in this partial system model and that is detected in the trace recordings and, thus, stored in

[1] `http://www.eclipse.org/app4mc`

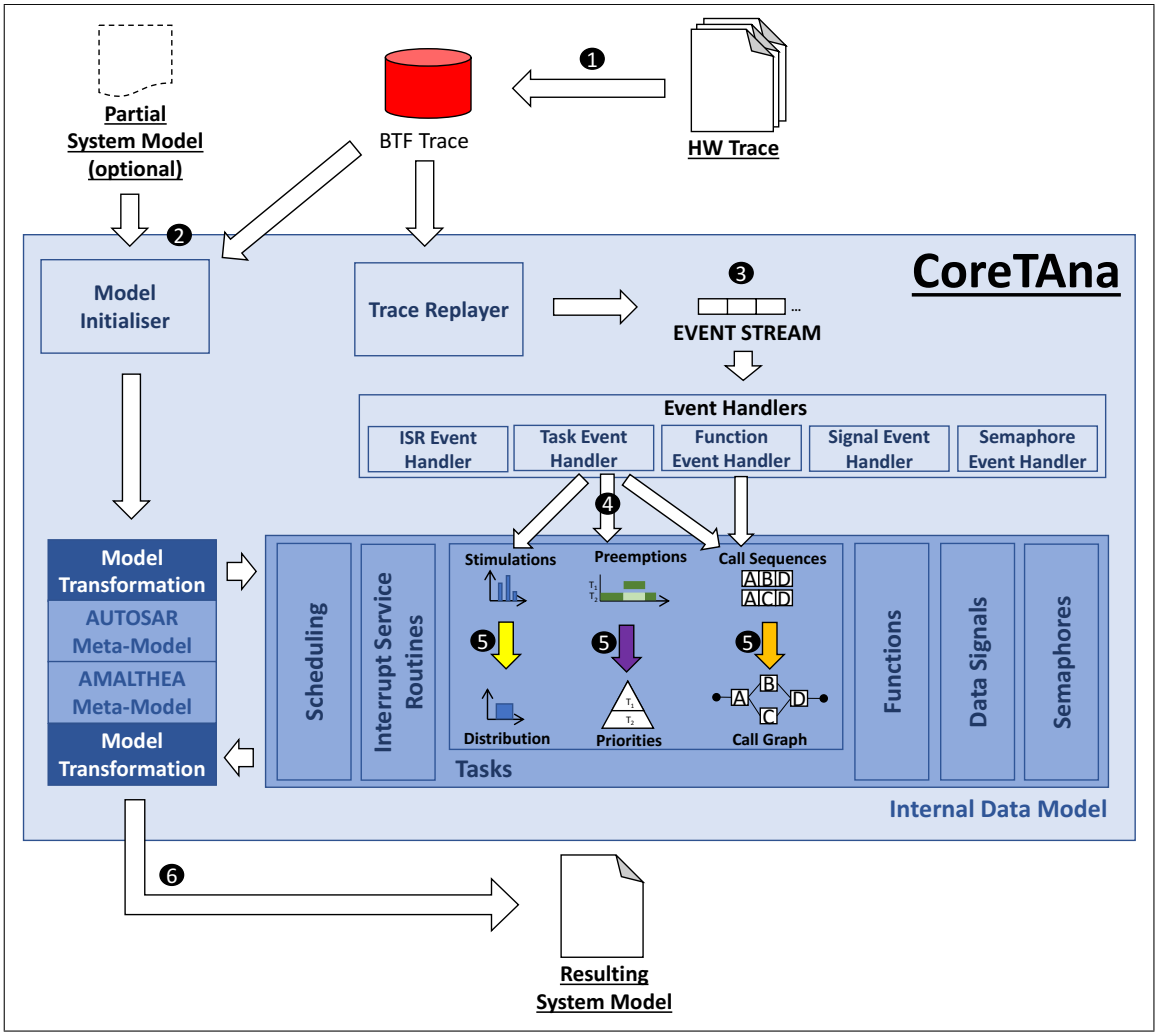


Figure 6.1.: CoreTAna's Software Design. The internal reverse engineering approach utilises the following technologies: SQLite, Eclipse Plug-In, Eclipse EMF, Eclipse MMT, Choco Constraint Solver, Apache Commons Math, Apache Commons Collections. Adapted from [15].

the databases, the *Model Initialiser* creates an object ❷. Each object implements the state machine of the entity's type according BTF and they altogether represent the *internal data model*.

The *Trace Replayer* creates for each trace recording an event stream such that all events are processed in chronological order ❸, which allows us to reason about event sequences such as context switches from one process to another. Each event triggers a transition in the state machine of the according object of the internal data model. To achieve a fast analysis, only the events that are relevant to the reverse engineering of a specific part of the model are considered, e.g., only the activation events are important for the stimulation-pattern use case, which is controlled by the event handler.

In the next step, the pieces of information collected from the trace events are processed ❹. The processing in chronological order allows us to detect coherences necessary to determine individual system characteristics such as pre-emptions ❺ (see Sec. 6.3). All obtained pieces of information from the trace recordings are analysed and the gained knowledge stored in the *internal data model*, which is, basically, a super set of all the information contained in the supported AUTOSAR-compliant models. This approach is necessary because of decisions that need to be made later on and that require a complete overview of the system's runtime behaviour such as the determination of a stimulation pattern from the individual task activations.

Finally, all synthesised information is transformed from the internal data model into an AUTOSAR-compliant model ❻ including AUTOSAR itself and AMALTHEA. In cases where there are multiple ways to model the observed behaviour, such as the representation of periodic activations by a sequence of single activations, the alternatives are assessed in the order of universality, starting with the most specific alternative as it can be seen in Algorithm 7.

As mentioned before, CoreTAna is part of the TA Tool Suite and is also included in the open-source tool chain APP4MC. Because both products are constructed on the Eclipse Rich Client Platform (RCP), CoreTAna is realised as an Eclipse Plug-in and utilises a number of Eclipse technologies. For example, the description of CoreTAna's internal data model is based on the Eclipse Modeling Framework (EMF). The decision for this is motivated by the fact that both models that can be generated by CoreTAna AUTOSAR and AMALTHEA are also based on EMF which makes the employment of model transformation possible for which we use the Query View Transformation (QVT) implementation of the Eclipse Model-to-Model Transformation (MMT) project.

Besides these Eclipse technologies which define mainly the architecture of our tool, CoreTAna utilises additional open-source libraries for the implementation of the reverse engineering algorithms. For example, some algorithms in Chapter 6.3 model a problem by stating a set of constraints that need to be satisfied. To be able to solve these constraint satisfaction problems (CSPs), we employ internally the Choco solver[2]. Other algorithms rely on the libraries of the Apache Commons project. Commons Math[3] helps us addressing our mathematical and statistical problems such as

[2] `http://www.choco-solver.org/`
[3] `http://commons.apache.org/proper/commons-math/`

goodness-of-fit tests and Commons Collections[4] provides us with powerful data structures such as the PATRICIA Trie that allow us to handle and analyse the vast amount of data that arises during the reverse engineering in an efficient way.

6.2. Approach

The reverse engineering approach of CoreTAna is designed in such a way that it fits seamlessly into the methodology specified by AUTOSAR. It also supports all the use cases that arise from our customer projects. For example, the most common challenge is to generate a model of a legacy software system, i.e., no model information is available at all. To do so, the customer provides us at first with a single trace recording that covers the general system behaviour but usually contains just events at process level because of technical limitations. This trace is then applied to CoreTAna as input which creates a rough frame of the system by determining the tasks and ISRs with their runtimes and the scheduling properties. With the help of DoTA, CoreTAna highlights also at once the deficits of the generated model which are inspected manually afterwards. Because our measure is composed of individual real-time metrics, the differences to the trace recording are easily comprehensible and allows one to identify the system parts that require additional information. Based on this frame and the results of the manual analysis, a first review of the model is done together with the customer in which the achieved quality and obvious deficits are discussed.

As an alternative, CoreTAna can also process multiple trace recordings in order to generate a model, e.g., if the trace recordings cover specific scenarios of the system's behaviour. In that case, each trace is transformed into a separate data base first. All these are then used by the model initialiser to create the internal model. After that, one trace after another is processed in chronological order and the observed events trigger the state machines of the internal objects. However, once a trace replay is completed, the objects are not analysed but their state machines are reset in order to continue with the next trace. After all data bases have been accessed, the gained knowledge stored in the internal model is analysed and transformed to a single AUTOSAR-compliant model.

Another common task for CoreTAna is the augmentation of a partially available model with additional information from a trace recording. This is, for example, the

[4] `http://commons.apache.org/proper/commons-collections/`

case if an initial model is already generated by a static analysis and has to be refined by the dynamic information of the system or if CoreTAna generated a model from a trace recording at process level which is gradually improved by analysing trace recordings at system level. To consider already existing knowledge in the reverse engineering approach, CoreTAna loads the information from the AUTOSAR-compliant model into its internal model. After that, the events of the new trace recording are processed and the gained information stored together with the existing knowledge in the internal objects. Finally, all these pieces of information are considered for the generation of a new AUTOSAR-compliant model which then not only covers the system's behaviour of the new trace recording but also that of the input model.

6.3. Algorithms

In Chapter 6.1, we introduced CoreTAna's software design which consists of the following three steps:

1. initialisation of the internal model (❷ in Fig. 6.1),
2. collecting information on the system by processing the events in the trace recording in chronological order (❸ and ❹),
3. and analysing and reasoning on the gained knowledge (❺).

This approach is also reflected in the design of the algorithms. As a consequence, the reverse engineering is also divided in three consecutive steps:

1. **Pre-processing:** The purpose of the pre-processing is to initialise the internal data model and, thus, it represents the implementation of the Model Initialiser as shown in Fig. 6.1. This includes determining all relevant entities in the system such as ISRs, tasks, runnables, data signals, and semaphores from the events of the trace recordings and requires that each entity can be identified by its unique name. For example, Algorithm 2 shows how all process entities are detected based on BTF events. Further, the allocation of the processes to a processing unit must be known before the trace can be processed in order to be able to analyse the scheduling.
2. **Processing:** Once all processes and their allocations are known, the information about each entity is refined further, for which, each trace recording that is available as input is processed in chronological order, one after another. Thereby, the internal data model is filled step-by-step with the information

Algorithm 1 Main Function

```
1:  function Main(trace)
2:      processes ← ∅                                  ▷ Set for process entities.
3:      runnables ← ∅                                  ▷ Set for runnable entities.
4:      allocations ← ()                               ▷ Tuple for mapping process to processing unit.
5:      activations ← [,]                              ▷ Two-dimensional matrix for activation moments.
6:      CSP_priorities ← ()                            ▷ Tuple for priority constraints and variables.
7:      preemptive ← ()                                ▷ Tuple for process pre-emptability.
8:      NETs ← [,]                                     ▷ Two-dimensional matrix for execution times.
9:      stimulations ← ()                              ▷ Tuple for stimulation pattern.
10:     priorities ← ()                                ▷ Tuple for process priorities.
11:     distributions ← ()                             ▷ Tuple for probability distributions.
12:     callSequences ← [,,]                           ▷ Three-dimensional matrix for call sequences.
13:     callGraph ← ∅                                  ▷ Empty set for branches with probability.
14:     ▷ Pre-Processing
15:     processes ← determineProcesses(trace)
16:     runnables ← determineRunnables(trace)
17:     allocations ← determineAllocation(trace)
18:     ▷ Processing
19:     activations ← prepareStimulation(trace)
20:     CSP_priorities ← preparePriorities(trace, allocations)
21:     preemptive ← determinePreemptability(trace, processes)
22:     NETs ← prepareExecutionTimes(trace)
23:     callSequences ← prepareCallSequences(trace)
24:     ▷ Post-Processing
25:     stimulations ← determineStimulation(activations, processes)
26:     priorities ← solve(CSP_priorities)
27:     distributions ← determineExecutionTimes(runnables, NETs)
28:     callGraph ← determineCallGraph(callSequences, processes)
29: end function
```

required for decisions that need to be made later on. In some cases, the determination of knowledge can already be done during the processing of a trace, e.g., information on the pre-emptability of a process.

3. **Post-processing:** Finally, all information collected in the internal data model is analysed and the missing elements for the AUTOSAR-compliant model are determined. This step is deliberately separated from the processing because some decisions that need to be made consider the information that is available on the system's runtime behaviour in total, i.e., all trace recordings must be processed first and not only one.

Algorithm 1 summarises the functionality of CoreTAna by listing all algorithms that are applied and by bundling them according to the aforementioned steps. A detailed description of each algorithm mentioned here is presented in the following sections.

Algorithm 2 Process Entities

Input: *trace* – Tuple containing all events in chronological order
Output: *processes* – Set containing the names of all process entities

```
1: function determineProcesses(trace)
2:     processes ← ∅                                   ▷ Set for processes entities.
3:     for all event in trace do
4:         if event[type] = 'T' or 'ISR' then
5:             if event[action] = 'activate' or 'start' or 'resume' or 'preempt' or 'wait' or 'release' or 'run' or
   'poll' or 'poll_parking' or 'park' or 'release_parking' then
6:                 add event[entity] to processes
7:     return processes
8: end function
```

6.3.1. OS Configuration

AUTOSAR [65] is defined as a multiprocessing operating system, i.e., multiple concurrent processes are executed in a system with each process running on a separate processing unit. Thus, the first pieces of information that have to be determined before a trace can be processed are the processes that are available in the system. This is described in Algorithm 2 where all process entities are determined based on BTF events. To do so, the individual elements of an event are accessed via their names given in Def. 5.1, e.g., *type*, *entity*, and *action* for the fourth, fifth, and seventh element, resp..

The algorithm shows that all processes are determined by going through the events of a trace recording and looking for task and ISR events, which can be identified by their type. The entity element of these events represents the name of a process according to the BTF specification. Alternatively, the same result can be achieved via a SQL statement similar to the following one, without the need to iterate over all events:

```
1 SELECT E.Name
2 FROM Entity as E, EntityType as T
3 WHERE (E.Type = T.ID) AND (T.Name = 'T' OR T.Name = 'ISR')
```

Listing 6.1: SQL statement for querying all process entities that occur in a trace recording.

Determining other system entities such as runnables is done analogously. After that, for each entity, an object is created that implements the state machine according to the entity's type.

Further, the allocation of the processes to a processing unit has to be known before

Algorithm 3 Allocation

Input: *trace* – Tuple containing all events in chronological order
Output: X – Tuple of variables representing the ID of the processing unit allocated by a process

```
1:  function determineAllocation(trace)
2:      runningState ← ∅                          ▷ Set of currently running processes.
3:      C ← ∅                                     ▷ Set of allocation constraints.
4:      X ← ()          ▷ Tuple of constraint variables representing the allocations.
5:      preempted ← 0                             ▷ Process that got preempted.
6:      for all event in trace do
7:          if event[type] = 'T' or 'ISR' then
8:              e ← event[entity]
9:              create constraint variable C_e
10:             X[e] ← C_e
11:             if event[action] = 'start' or 'resume' or 'run' then
12:                 add constraint to C: X[e] = X[preempted]
13:                 preempted ← 0
14:                 for all process in runningState do
15:                     add constraint to C: X[e] ≠ X[process]
16:                 add e to runningState
17:             else
18:                 remove e from runningState
19:                 if event[action] = 'preempt' then
20:                     preempted ← e
21:     CSP ← (X, C)
22:     ▷ Find an allocation by solving the CSP for the given constraints.
23:     solve(CSP)
24:     return X
25: end function
```

the trace can be processed in order to detect pre-emptions. Because AUTOSAR applies local scheduling, where each processing unit is managed by its own scheduler, the allocations can be determined according to Algorithm 3. For this, let C_p denote a constraint variable that represents the ID of the processing unit allocated by process p. Due to the fact that processing units are shared resources, which can only be occupied by one process at a time, the process events are processed in chronological order to detect concurrent executions and pre-emptions. These pieces of information are used to define a Constraint Satisfaction Problem (CSP) (line 12 and 15 of Algorithm 3), which then yields an allocation of process to processing unit that fulfils the observations recorded in the underlying trace.

Example 1 (Allocation)**.** In order to show how the algorithms work, a short example is introduced in the following. It describes a simple system which consists of three tasks (Task_1, Task_2, and Task_3) and runs on two processing units (CORE0 and CORE1). Task_1 and Task_2 are scheduled by the operating system on CORE0 and Task_3 is mapped to CORE1. Furthermore, Task_1 is known to call Runnable_1, Runnable_2,

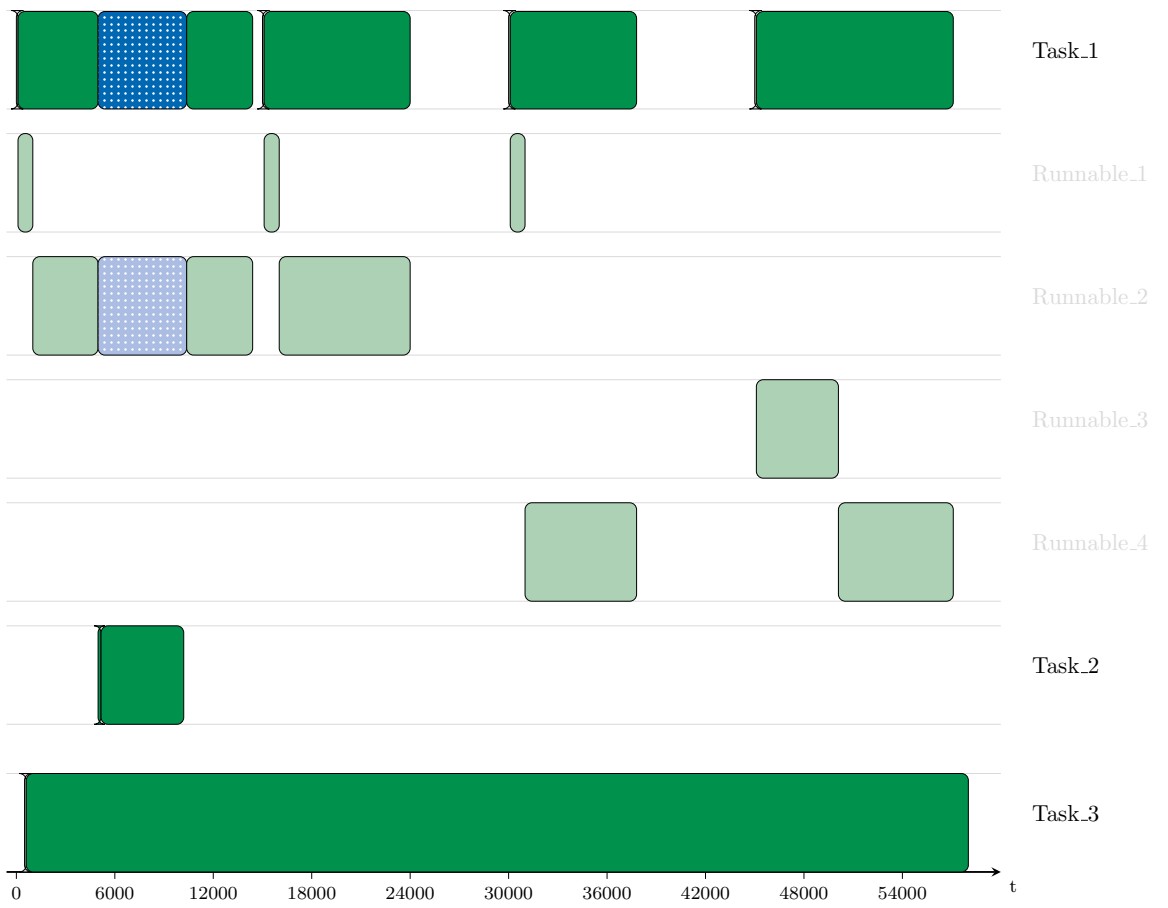


Figure 6.2.: Gantt Chart of Trace Example for Showing CoreTAna's Algorithms. Visualisation of the trace recording described in Example 1 in a Gantt chart.

Runnable_3, and Runnable_4 and is activated periodically every 10 ms.

Fig. 6.2 visualises a trace recording that covers 58 ms of this system's internal behaviour in a Gantt chart. The according BTF trace is shown in Listing 6.2. There, also the intermediate states of the most relevant variables used in Algorithm 3 are added in red in order to illustrate how this algorithm determines the allocation of a process to a processing unit.

```
0,CORE0,0,T,Task_1,0,activate
  X = (C_Task1), runningState = ∅
100,CORE0,0,T,Task_1,0,start
  runningState = {Task1}
100,Task_1,0,R,Runnable_1,0,start
500,CORE1,0,T,Task_3,0,activate
  X = (C_Task1, C_Task3)
600,CORE1,0,T,Task_3,0,start
```

```
   runningState = {Task_1, Task_3}; C = {C_Task_1 ≠ C_Task_3}
10 1000,Task_1,0,R,Runnable_1,0,terminate
   1000,Task_1,0,R,Runnable_2,0,start
12 5000,CORE0,0,T,Task_2,0,activate
   X = (C_Task_1, C_Task_3, C_Task_2)
14 5200,Task_1,0,R,Runnable_2,0,suspend
   5200,CORE0,0,T,Task_1,0,preempt
16   runningState = {Task_3}, preempted = Task_1
   5200,CORE0,0,T,Task_2,0,start
18   runningState = {Task_3, Task_2}; preempted = 0; C = {C_Task_1 ≠ C_Task_3, C_Task_2 =
       C_Task_1}
   10200,CORE0,0,T,Task_2,0,terminate
20   runningState = {Task_3}
   10400,CORE0,0,T,Task_1,0,resume
22   runningState = {Task_3, Task_1}
   10400,Task_1,0,R,Runnable_2,0,resume
24 14400,Task_1,0,R,Runnable_2,0,terminate
   14400,CORE0,0,T,Task_1,0,terminate
26   runningState = {Task_3}
   15000,CORE0,0,T,Task_1,1,activate
28 15100,CORE0,0,T,Task_1,1,start
   runningState = {Task_3, Task_1}
30 15100,Task_1,1,R,Runnable_1,1,start
   16000,Task_1,1,R,Runnable_1,1,terminate
32 16000,Task_1,1,R,Runnable_2,1,start
   24200,Task_1,1,R,Runnable_2,1,terminate
34 24200,CORE0,0,T,Task_1,1,terminate
   runningState = {Task_3}
36 30000,CORE0,0,T,Task_1,2,activate
   30100,CORE0,0,T,Task_1,2,start
38   runningState = {Task_3, Task_1}
   30100,Task_1,2,R,Runnable_1,2,start
40 31000,Task_1,2,R,Runnable_1,2,terminate
   31000,Task_1,2,R,Runnable_4,0,start
42 38000,Task_1,2,R,Runnable_4,0,terminate
```

```
   38000,CORE0,0,T,Task_1,2,terminate
44   runningState = {Task_3}
   45000,CORE0,0,T,Task_1,3,activate
46 45100,CORE0,0,T,Task_1,3,start
   runningState = {Task_3, Task_1}
48 45100,Task_1,3,R,Runnable_3,0,start
   50100,Task_1,3,R,Runnable_3,0,terminate
50 50100,Task_1,3,R,Runnable_4,1,start
   57100,Task_1,3,R,Runnable_4,1,terminate
52 57100,CORE0,0,T,Task_1,3,terminate
   runningState = {Task_3}
54 58000,CORE1,0,T,Task_3,0,terminate
   runningState = ∅, C_Task1 = 0, C_Task2 = 0, C_Task3 = 1
```

Listing 6.2: Example trace for showing CoreTAna's algorithm for determining the allocation.

The most relevant lines in Listing 6.2 for comprehending the algorithm are the lines in which the constraints are defined. In line 8, constraint "$C_{Task_1} \neq C_{Task_3}$" is created because of the knowledge that both tasks $Task_1$ and $Task_3$ are running at that moment, i.e., they must be allocated to different processing units. The other constraint "$C_{Task_2} = C_{Task_1}$" is added to the constraint satisfaction problem as a result of line 17. At that point in time it is known that $Task_1$ is pre-empted by $Task_2$. This fact allows one to conclude that both tasks must run on the same processing unit. And this is also what the constraint solver yields with $C_{Task_1} = 0, C_{Task_2} = 0, C_{Task_3} = 1$ in line 55. The resulting integers for the constraint variable state the allocated processing unit, i.e., $Task_1$ and $Task_2$ execute on the same processing unit because they have the same number and $Task_3$ on a different one.

6.3.2. Scheduling Properties

The temporal behaviour of a real-time system is partially determined by the employed scheduling policy. Therefore, it is crucial to reversely engineer the system's scheduling parameters, which includes in case of an AUTOSAR-compliant system the process priorities and the information whether a process can be pre-empted or not.

The AUTOSAR OS provides a fixed priority-based scheduling policy [65, p. 83 f.], which means that the scheduler assigns a managed resource to the process with the

Algorithm 4 Preparation of Process Priority Determination

Input: *trace* – Tuple containing all events in chronological order
processes – Set containing the names of all process entities
allocations – Set containing the ID of the processing unit allocated by a process
Output: *CSP* – Tuple containing a set of variables that represents the priorities of the processes and a set of constraints that puts the priorities in relation to

```
1: function preparePriorities(trace, allocations)
2:     readyState ← ()                          ▷ Tuple with all process instances in state ready.
3:     C ← ∅                                    ▷ Set of priority constraints.
4:     X ← ()                                   ▷ Tuple of constraint variables representing a process priority.
5:     for all event in trace do
6:         if event[type] = 'T' or 'ISR' then
7:             e ← event[entity]
8:             create constraint variable C_e
9:             X[e] ← C_e
10:            instance ← event[entityInstance]
11:            allocation ← allocations[e]
12:            if event[action] = 'start' or 'resume' then
13:                remove (e, instance) from readyState[allocation]
14:                for all process in readyState[allocation] do
15:                    if age(e) < age(process) then
16:                        add constraint to C: X[e] > X[process]
17:                    else
18:                        add constraint to C: X[e] ≥ X[process]
19:            else if event[action] = 'activate' or 'preempt' or 'park' then
20:                add (e, instance) to readyState[allocation]
21:    CSP ← (X, C)
22:    return CSP
23: end function
```

highest priority. The priority of each process is set before runtime and does not change during execution. It is represented by a positive numerical value, which "has to be understood as a relative value, i.e. the values show only the relative ordering of the [processes]" [65, p. 227].

The priority ordering of processes can be determined from trace events according Algorithm 4 where $age(x_i)$ denotes a function that yields the time since the activation of process instance x_i. Based on the knowledge that processes with higher priorities are preferred, each process execution reveals the process with the highest priority among all processes that are ready to be scheduled on a processing unit at the observed point in time (see line 11 ff. in Algorithm 4). In order to do so, the allocation, i.e., the processing units on which an activated process can execute, has to be known. If the activation moment of the started processes dates back furthest, the priority relation is clear because of the fact that processes with the same priority on the same processing unit are executed in order of their activation (see line 15). Using this knowledge, the trace recording is processed and a constraint describing

Algorithm 5 Process Pre-emptability

Input: *trace* – Tuple containing all events in chronological order
processes – Set containing the names of all process entities
Output: *preemptive* – Tuple containing the pre-emptability of each process

```
1:  function determinePreemptability(trace, processes)
2:      preemptive ← ()                          ▷ Tuple for process pre-emptability.
3:      for all process in processes do
4:          preemptive[process] ← 'NON'
5:      for all event in trace do
6:          if event[type] = 'T' or 'ISR' then
7:              if event[action] = 'preempt' or 'park' then
8:                  e ← event[entity]
9:                  preemptive[e] ← 'FULL'
10:     return preemptive
11: end function
```

the priority relation between two processes is added to the CSP if such an according situation is detected. After all trace recordings are processed, the resulting CSP is solved in the post-processing step and, for each process, the solver yields a priority that satisfies the behaviour observed in the trace.

Another scheduling property that drives the scheduling policy of the AUTOSAR OS is the pre-emptability of a process [65, p. 228]. Pre-emptability defines whether the execution of a process can be suspended, e.g., in order to execute a higher prioritised process. The AUTOSAR OS supports mixed pre-emptive systems, which means that the software is made up of full pre-emptive and non pre-emptive processes in any combination. Algorithm 5 describes a way to analyse a trace recording regarding the pre-emptive characteristics of its processes. The algorithm assumes that, in general, all processes are non pre-emptive (see line 4 in Algorithm 5). If a pre-emption is detected in the trace, the pre-emptive property of the process is changed to full pre-emptive (see lines 6 ff.). Thus, this algorithm yields its results already in the processing step.

Example 2 (Scheduling Properties). In order to show how the algorithms that are presented in this section work, we continue with the simple system introduced in Example 1. Listing 6.2 depicts the BTF trace recording that covers 58 ms of this system's internal behaviour. Again, the intermediate states of the most relevant variables used in Algorithm 4 and Algorithm 5 are added in red to illustrate how these algorithms determine the process priorities and whether a process can be pre-empted.

```
   preemptive = (Task_1 = 'NON', Task_2 = 'NON', Task_3 = 'NON')
2 0,CORE0,0,T,Task_1,0,activate
```

```
   X = (C_Task1); readyState[0] = (Task1, 0)
4  100, CORE0, 0, T, Task_1, 0, start
   readyState = ()
6  100, Task_1, 0, R, Runnable_1, 0, start
   500, CORE1, 0, T, Task_3, 0, activate
8     X = (C_Task1, C_Task3); readyState[1] = (Task3, 0)
   600, CORE1, 0, T, Task_3, 0, start
10    readyState = ()
   1000, Task_1, 0, R, Runnable_1, 0, terminate
12 1000, Task_1, 0, R, Runnable_2, 0, start
   5000, CORE0, 0, T, Task_2, 0, activate
14    X = (C_Task1, C_Task3, C_Task2); readyState[0] = (Task2, 0)
   5200, Task_1, 0, R, Runnable_2, 0, suspend
16 5200, CORE0, 0, T, Task_1, 0, preempt
   preemptive[Task1] = 'FULL'; readyState[0] = {(Task1, 0), (Task2, 0)}
18 5200, CORE0, 0, T, Task_2, 0, start
   age(Task1) = 5200; age(Task2) = 200; C = {C_Task2 > C_Task1}
20 10200, CORE0, 0, T, Task_2, 0, terminate
   10400, CORE0, 0, T, Task_1, 0, resume
22    readyState = ()
   10400, Task_1, 0, R, Runnable_2, 0, resume
24 14400, Task_1, 0, R, Runnable_2, 0, terminate
   14400, CORE0, 0, T, Task_1, 0, terminate
26 15000, CORE0, 0, T, Task_1, 1, activate
   readyState[0] = (Task1, 1)
28 15100, CORE0, 0, T, Task_1, 1, start
   readyState = ()
30 15100, Task_1, 1, R, Runnable_1, 1, start
   16000, Task_1, 1, R, Runnable_1, 1, terminate
32 16000, Task_1, 1, R, Runnable_2, 1, start
   24200, Task_1, 1, R, Runnable_2, 1, terminate
34 24200, CORE0, 0, T, Task_1, 1, terminate
   30000, CORE0, 0, T, Task_1, 2, activate
36    readyState[0] = (Task1, 2)
   30100, CORE0, 0, T, Task_1, 2, start
```

```
38    readyState = ()
      30100,Task_1,2,R,Runnable_1,2,start
40    31000,Task_1,2,R,Runnable_1,2,terminate
      31000,Task_1,2,R,Runnable_4,0,start
42    38000,Task_1,2,R,Runnable_4,0,terminate
      38000,CORE0,0,T,Task_1,2,terminate
44    45000,CORE0,0,T,Task_1,3,activate
      readyState[0] = (Task_1,3)
46    45100,CORE0,0,T,Task_1,3,start
      readyState = ()
48    45100,Task_1,3,R,Runnable_3,0,start
      50100,Task_1,3,R,Runnable_3,0,terminate
50    50100,Task_1,3,R,Runnable_4,1,start
      57100,Task_1,3,R,Runnable_4,1,terminate
52    57100,CORE0,0,T,Task_1,3,terminate
      58000,CORE1,0,T,Task_3,0,terminate
54    C_Task_1 = 0, C_Task_2 = 1, C_Task_3 = 0
```

Listing 6.3: Example trace for showing CoreTAna's algorithms for determining the scheduling properties

The most relevant lines in Listing 6.2 for comprehending the algorithms are line 16 for Algorithm 5 and line 18 for Algorithm 4. The former trace event represents a pre-emption of $Task_1$ which results in the fact that $Task_1$ is stated as pre-emptive. Furthermore, it has the effect that $Task_1$ returns to the processing state ready in that moment. This is of crucial importance for Algorithm 4 because the next trace event starts the execution of $Task_2$. Due to the fact that $Task_1$ was activated before $Task_2$, the constraint "$C_{Task_2} > C_{Task_1}$" is is added to the constraint satisfaction problem. Finally, the constraint solver yields a possible priority assignment for each process in line 54, which respects the determined constraint that the priority of $Task_2$ has to be higher than that of $Task_1$.

6.3.3. Stimulation

Besides the parameters of the system's scheduling policy, the instants in time at which a process is activated must be identified in order to determine the temporal behaviour of a real-time system [3]. To make a statement about the activation pattern

of a process, at first, the activation moments for each process are determined from the trace recording as described by Algorithm 6. In the processing step, the timestamps are stored for later analysis in a two-dimensional matrix called *activations[,]* where each row vector contains the activation moments of a specific process.

After all trace recordings are processed and all activation moments of the processes are collected in the internal data model, the gathered information is analysed in the post-processing step to generate matching activation patterns. AUTOSAR provides two ways for specifying temporal activation patterns for tasks: *OSAlarm* [65, p. 197 ff.] and *OSScheduleTable* [65, p. 220 ff.]. An alarm describes a periodic activation and is defined by the *AlarmTime*, which sets the relative or absolute time when the alarm expires for the first time, and the *CycleTime*, which determines the time until the alarm's recurrence. A schedule table is an encapsulation for a set of so-called *ExpiryPoints*, which define a recurring sequence of activations. Each expiry point specifies a point in time relative to the start of the schedule table at which the OS activates a task. The schedule table itself is determined by an absolute or relative offset and a duration, which sets the period for a cyclic repetition.

In contrast to AUTOSAR, AMALTHEA[57] allows one to define activation patterns also for ISRs. In AMALTHEA, a single activation is called *SingleStimulus*, a periodic activation is modelled with the help of the *PeriodicStimulus* pattern and, for non-periodic activations, the *PeriodicSnytheticStimulus* is used without setting a recurrence. The semantics of these terms are equal to those of their counterparts in AUTOSAR which is why they are determined the same way. In addition, AMALTHEA provides a stimulation pattern called *Arrival Curve* that is an alternative way to model periodic and non-periodic activations. For these reasons, we enable the gen-

Algorithm 6 Preparation of Stimulation Pattern Determination

Input: *trace* – Tuple that contains all events in chronological order
Output: *activations* – Two-dimensional matrix that contains all activation moments for each process

```
1:  function prepareStimulation(trace)
2:      activations ← [,]                    ▷ Two-dimensional matrix of process activations.
3:      for all event in trace do
4:          if event[type] = 'T' or 'ISR' then
5:              if event[action] = 'activate' then
6:                  e ← event[entity]
7:                  instance ← event[entityInstance]
8:                  activations[e, instance] ← event[timestamp]
9:      return activations
10: end function
```

Algorithm 7 Determination of Stimulation Patterns

Input: *activations* – Two-dimensional matrix containing all activation moments of each process
processes – Set containing the names of all process entities
Output: *stimulations* – Tuple of stimulation patterns describing the temporal activations for each process

```
1:  function determineStimulation(activations, processes)
2:      stimulations ← ()                                  ▷ Tuple for stimulation patterns.
3:      for process in processes do
4:          if length(activations[process]) = 1 then
5:              ▷ Single activation
6:              alarm ← ()                                 ▷ Tuple for stimulation information.
7:              alarm[AlarmTime] ← activations[process, 0]
8:              alarm[CycleTime] ← 0
9:              stimulations[process] ← alarm
10:         else
11:             a2a ← ()                                   ▷ Tuple for inter-arrival times.
12:             for i ← 0 to length(activations[process]) - 2 do
13:                 a2a[i] ← activations[process, i + 1] − activations[process, i]
14:             if sd(a2a) / mean(a2a) < 0.5 % then
15:                 ▷ Periodic activation
16:                 alarm ← ()                             ▷ Tuple for stimulation information.
17:                 alarm[AlarmTime] ← activations[process, 0]
18:                 alarm[CycleTime] ← mean(a2a)
19:                 stimulations[process] ← alarm
20:             else
21:                 ▷ Non-periodic activation
22:                 scheduleTable ← ()                     ▷ Tuple for stimulation information.
23:                 for i ← 0 to length(activations[process]) - 1 do
24:                     expiryPoint ← ()
25:                     expiryPoint[offset] ← activations[process, i]
26:                     scheduleTable[i] ← expiryPoint
27:                 stimulations[process] ← scheduleTable
28:     return stimulations
29: end function
```

eration of that pattern via an additional input parameter for our reverse engineering algorithm.

Let *mean* and *sd* be functions that yield the mean and standard deviation, resp., from a tuple. Then, the stimulation of a process can be identified as a periodic activation (*Alarm*) or a sequence of activations (*ScheduleTable*) from trace events as described by Algorithm 7. Due to the fact that there are multiple ways to model the observed stimulation behaviour, such as the representation of periodic activations by a sequence of single activations, the alternatives are assessed in the order of universality, starting with the most specific alternative. For each process, the algorithm checks first whether there is only a single activation recorded in the trace (see lines 5 ff. in Algorithm 7). If this is the case, a non-recurring alarm is created and associated with

the process. Otherwise, the algorithm continues and checks whether the observed activations are roughly periodic, by determining the standardized moment of the resulting distribution of the process's inter-arrival times. A moment below 0.5 %, which is a common significance level for statistical tests [71, p. 30], confirms that the process is activated by a cyclic alarm (see lines 14 ff.). Finally, if the observed activations are found to be neither singular nor periodic, the individual activation times are transferred into expiry points of a schedule table (see lines 22 ff.).

Example 3 (Stimulation)**.** Again, we use the simple system introduced in Example 1 in order to show how the algorithms work. Listing 6.4 depicts the BTF trace recording that covers 58 ms of this system's internal behaviour. The intermediate states of the most relevant variables used in Algorithm 6 are added in red to illustrate how this algorithm gathers the information for determining the stimulation later on via Algorithm 7.

```
0,CORE0,0,T,Task_1,0,activate
    activations[Task_1] = (0)
100,CORE0,0,T,Task_1,0,start
100,Task_1,0,R,Runnable_1,0,start
500,CORE1,0,T,Task_3,0,activate
    activations[Task_3] = (500)
600,CORE1,0,T,Task_3,0,start
1000,Task_1,0,R,Runnable_1,0,terminate
1000,Task_1,0,R,Runnable_2,0,start
5000,CORE0,0,T,Task_2,0,activate
    activations[Task_2] = (5000)
5200,Task_1,0,R,Runnable_2,0,suspend
5200,CORE0,0,T,Task_1,0,preempt
5200,CORE0,0,T,Task_2,0,start
10200,CORE0,0,T,Task_2,0,terminate
10400,CORE0,0,T,Task_1,0,resume
10400,Task_1,0,R,Runnable_2,0,resume
14400,Task_1,0,R,Runnable_2,0,terminate
14400,CORE0,0,T,Task_1,0,terminate
15000,CORE0,0,T,Task_1,1,activate
    activations[Task_1] = (0,15000)
15100,CORE0,0,T,Task_1,1,start
```

```
   15100,Task_1,1,R,Runnable_1,1,start
24 16000,Task_1,1,R,Runnable_1,1,terminate
   16000,Task_1,1,R,Runnable_2,1,start
26 24200,Task_1,1,R,Runnable_2,1,terminate
   24200,CORE0,0,T,Task_1,1,terminate
28 30000,CORE0,0,T,Task_1,2,activate
     activations[Task_1] = (0,15000,30000)
30 30100,CORE0,0,T,Task_1,2,start
   30100,Task_1,2,R,Runnable_1,2,start
32 31000,Task_1,2,R,Runnable_1,2,terminate
   31000,Task_1,2,R,Runnable_4,0,start
34 38000,Task_1,2,R,Runnable_4,0,terminate
   38000,CORE0,0,T,Task_1,2,terminate
36 45000,CORE0,0,T,Task_1,3,activate
     activations[Task_1] = (0,15000,30000,45000)
38 45100,CORE0,0,T,Task_1,3,start
   45100,Task_1,3,R,Runnable_3,0,start
40 50100,Task_1,3,R,Runnable_3,0,terminate
   50100,Task_1,3,R,Runnable_4,1,start
42 57100,Task_1,3,R,Runnable_4,1,terminate
   57100,CORE0,0,T,Task_1,3,terminate
44 58000,CORE1,0,T,Task_3,0,terminate
```

Listing 6.4: Example trace for showing CoreTAna's algorithms for determining the stimulation patterns

Gathering all the information that is necessary for Algorithm 7 to determine a suitable stimulation pattern is straight forward. Every time an activation event appears in the trace recording, the timestamp of that event is stored in a data structure under the respective entity. This data structure is then used as input for Algorithm 7 which yields, for example, a periodic stimulation pattern in case of $Task_1$. The intermediate states of the most relevant variables in order to comprehend the algorithm are stated in the following:

$length(activations[Task_1]) = 4$

$a2a[0] = activations[Task_1][1] - activations[Task_1][0] = 15000 - 0 = 15000$

$a2a[1] = activations[Task_1][2] - activations[Task_1][1] = 30000 - 15000 = 15000$

$$a2a[2] = activations[Task_1][3] - activations[Task_1][2] = 45000 - 30000 = 15000$$
$$mean(a2a) = \frac{15000+15000+15000}{3} = 15000$$
$$sd(a2a) = \sqrt{\frac{1}{3}\{(15000-15000)^2 + (15000-15000)^2 + (15000-15000)^2\}} = 0$$
$$\frac{sd(a2a)}{mean(a2a)} = \frac{0}{15000} = 0$$
$$alarm[AlarmTime] = 0$$
$$alarm[CycleTime] = 15000$$

Algorithm 8 Prepare Execution Times

Input: *trace* – Tuple containing all events in chronological order
Output: *NETs* – Two-dimensional matrix containing all observed net execution times of the runnables and processes

```
1:  function prepareExecutionTimes(trace)
2:      GETs ← [,]                          ▷ Two-dimensional matrix of gross execution times.
3:      NETs ← [,]                          ▷ Two-dimensional matrix of net execution times.
4:      starts ← [,]                        ▷ Two-dimensional matrix of start instants.
5:      suspends ← [,]                      ▷ Two-dimensional matrix of suspend instants.
6:      resumes ← [,]                       ▷ Two-dimensional matrix of resume instants.
7:      terminates ← [,]                    ▷ Two-dimensional matrix of terminate instants.
8:      entities ← ∅                        ▷ Set of all process and runnable entities.
9:      for all event in trace do
10:         if event[type] = 'T' or 'ISR' or 'R' then
11:             e ← event[entity]
12:             instance ← event[entityInstance]
13:             add e to entities
14:             if event[action] = 'start' then
15:                 starts[e, instance] ← event[timestamp]
16:             else if event[action] = 'suspend' or 'preempt' then
17:                 suspends[e, instance] ← event[timestamp]
18:             else if event[action] = 'resume' then
19:                 resumes[e, instance] ← event[timestamp]
20:             else if event[action] = 'terminate' then
21:                 terminates[e, instance] ← event[timestamp]
22:     for e in entities do
23:         for i ← 0 to length(starts[e]) - 1 do
24:             GETs[e, i] ← terminates[e, i] − starts[e, i]
25:             NETs[e, i] ← GETs[e, i]
26:             for j ← 0 to length(suspends[e]) - 1 do
27:                 if start[e, i] < suspends[e, j] and
                      resumes[entity, j] < terminates[e, i] then
28:                     preemption ← resumes[e, j] − suspends[e, j]
29:                     NETs[e, i] ← NETs[e, i] − preemption
30:     return NETs
31: end function
```

6.3.4. Runtime Behaviour

Another property that determines the temporal behaviour of a real-time system is the execution time of a process. AUTOSAR only provides a means to specify the *Resource Consumption* [72, p. 112 ff.] of a *Runnable Entity*. Because runnables are called within the context of a process, CoreTAna derives the execution times of a runnable entity from the temporal behaviour of the process, in case no detailed information on individual runnables is contained within a trace recording. This means that the runtime behaviour of a process is represented by at least a single runnable entity. The resource consumption is specified by the gross execution time (GET) and net execution time (NET) of a runnable entity. Their values can be determined from a trace recording, as shown in Algorithm 8.

The GET is defined as the time interval between the start and the termination of a process or runnable (see line 24 in Algorithm 8). In contrast, the NET states the amount of time during which this entity actually executes on a processing core and, thus, does not include the time during which the entity is pre-empted (see lines 28 f.).

In the processing step, CoreTAna only collects the individual times for each entity from one or multiple trace recordings. Based on this data, the observable runtime behaviour is then quantitatively summarised in the post-processing step. To do so, AUTOSAR provides statistical measures such as minimum, maximum, and nominal for each execution time. AMALTHEA even allows one to model the probability of the execution time values with the help of the following five statistical distributions (see Sec. 4.2.2): constant value, uniform distribution, normal distribution, Weibull distribution [51], beta distribution [52].

Thus, the main goal of the post-processing step is to find a probability distribution that fits the sampledprobability distribution best. This is done with the help of the Kolmogorov-Smirnov Test (K-S Test) [71, p. 192 ff.], which is a statistical goodness-of-fit test that summarises the differences between observed values and the values expected from a given probability distribution. At first, we determine the parameters of each supported probability distribution such as the mean and the standard deviation for the normal distribution from the sample values (see lines 19 f. in Algorithm 9). After that, the K-S Test calculates the significance level that states the probability that the null hypothesis, namely that the observed sample values follow the distribution specified by the parameters, is rejected. Thus, the probability distribution that fits best to the sample with the reference probability distribution is identified by the low-

Algorithm 9 Determine Execution Time Distribution

Input: *entities* – Set containing the names of all runnable entities
NETs – Two-dimensional matrix containing all observed net execution times of the runnables and processes
Output: *distributions* – Tuple containing definitions of the probability distributions for each entity

```
1: function determineExecutionTime(entities, NETs)
2:     distributions ← ()                                   ▷ Tuple for the distributions.
3:     for all entity in entities do
4:         values ← NETs[entity]
5:         if length(values) = 1 then                       ▷ Constant
6:             distributions[entity] ← constant(NETs[entity, 0])
7:         else
8:             value ← mostFrequentValue(values)            ▷ Constant
9:             constant ← constant(value)
10:            minSL ← kolmogorovSmirnovTest(constant, values)
11:            distributions[entity] ← constant
12:            min ← min(values)                            ▷ Uniform distribution
13:            max ← max(values)
14:            uni ← uniform(min, max)
15:            significanceLevel ← kolmogorovSmirnovTest(uni, values)
16:            if significanceLevel< minSL then
17:                minSL ←significanceLevel
18:                distributions[entity] ← uni
19:            mean ← mean(values)                          ▷ Normal distribution
20:            sd ← sd(values)
21:            normal ← normal(mean, sd)
22:            significanceLevel ← kolmogorovSmirnovTest(normal, values)
23:            if significanceLevel< minSL then
24:                minSL ←significanceLevel
25:                distributions[entity] ← normal
26:            lambda ← lambda(values)                      ▷ Weibull distribution
27:            kappa ← kappa(values)
28:            weibull ← weibull(lambda, kappa)
29:            significanceLevel ← kolmogorovSmirnovTest(weibull, values)
30:            if significanceLevel < minSL then
31:                minSL ← significanceLevel
32:                distributions[entity] ← weibull
33:            α ← alpha(values)                            ▷ Beta distribution
34:            β ← beta(values)
35:            beta ← B(α, β)
36:            significanceLevel ← kolmogorovSmirnovTest(beta, values)
37:            if significanceLevel < minSL then
38:                distributions[entity] ← beta
39:        return distributions
40: end function
```

est significance level resulting from the K-S Test (see, e.g., lines 23 ff.).

Example 4 (Execution Time). Listing 6.5 depicts the BTF trace recording of the simple system that was introduced first in Example 1. It covers 58 ms of the system's internal behaviour and contains in red the intermediate states of the most relevant variables

used in Algorithm 8 to gather the information for determining an execution time distribution later on via Algorithm 9.

```
0,CORE0,0,T,Task_1,0,activate
100,CORE0,0,T,Task_1,0,start
  starts[Task_1] = (100)
100,Task_1,0,R,Runnable_1,0,start
  starts[Runnable_1] = (100)
500,CORE1,0,T,Task_3,0,activate
600,CORE1,0,T,Task_3,0,start
  starts[Task_3] = (600)
1000,Task_1,0,R,Runnable_1,0,terminate
  terminates[Runnable_1] = (1000)
1000,Task_1,0,R,Runnable_2,0,start
  starts[Runnable_2] = (1000)
5000,CORE0,0,T,Task_2,0,activate
5200,Task_1,0,R,Runnable_2,0,suspend
  suspends[Runnable_2] = (5200)
5200,CORE0,0,T,Task_1,0,preempt
  suspends[Task_1] = (5200)
5200,CORE0,0,T,Task_2,0,start
  starts[Task_2] = (5200)
10200,CORE0,0,T,Task_2,0,terminate
  terminates[Task_2] = (10200)
10400,CORE0,0,T,Task_1,0,resume
  resumes[Task_1] = (10400)
10400,Task_1,0,R,Runnable_2,0,resume
  resumes[Runnable_2] = (10400)
14400,Task_1,0,R,Runnable_2,0,terminate
  terminates[Runnable_2] = (14400)
14400,CORE0,0,T,Task_1,0,terminate
  terminates[Task_1] = (14400)
15000,CORE0,0,T,Task_1,1,activate
15100,CORE0,0,T,Task_1,1,start
  starts[Task_1] = (100,15100)
15100,Task_1,1,R,Runnable_1,1,start
```

```
34    starts[Runnable_1] = (100,15100)
  16000,Task_1,1,R,Runnable_1,1,terminate
36    terminates[Runnable_1] = (1000,16000)
  16000,Task_1,1,R,Runnable_2,1,start
38    starts[Runnable_2] = (1000,16000)
  24200,Task_1,1,R,Runnable_2,1,terminate
40    terminates[Runnable_2] = (144000,24200)
  24200,CORE0,0,T,Task_1,1,terminate
42    terminates[Task_1] = (14400,24200)
  30000,CORE0,0,T,Task_1,2,activate
44  30100,CORE0,0,T,Task_1,2,start
      starts[Task_1] = (100,15100,30100)
46  30100,Task_1,2,R,Runnable_1,2,start
      starts[Runnable_1] = (100,15100,30100)
48  31000,Task_1,2,R,Runnable_1,2,terminate
      termiantes[Runnable_1] = (1000,16000,31000)
50  31000,Task_1,2,R,Runnable_4,0,start
      starts[Runnable_4] = (31000)
52  38000,Task_1,2,R,Runnable_4,0,terminate
      terminates[Runnable_4] = (38000)
54  38000,CORE0,0,T,Task_1,2,terminate
      termiantes[Task_1] = (14400,24000,38000)
56  45000,CORE0,0,T,Task_1,3,activate
  45100,CORE0,0,T,Task_1,3,start
58    starts[Task_1] = (100,15100,30100,45100)
  45100,Task_1,3,R,Runnable_3,0,start
60    starts[Runnable_3] = (45100)
  50100,Task_1,3,R,Runnable_3,0,terminate
62    starts[Task_1] = (50100)
  50100,Task_1,3,R,Runnable_4,1,start
64    starts[Task_1] = (31000,50100)
  57100,Task_1,3,R,Runnable_4,1,terminate
66    terminates[Runnable_4] = (38000,57100)
  57100,CORE0,0,T,Task_1,3,terminate
68    termiantes[Task_1] = (14400,24000,38000,57100)
```

```
58000,CORE1,0,T,Task_3,0,terminate
70  terminates[Task_3] = (58000)
```

Listing 6.5: Example trace for showing CoreTAna's algorithms for determining the execution time distributions

Listing 6.4 shows the gathering of all the information required for determining an execution time distribution. It is done straight forward by storing the timestamp of every state change event of a process from or to running state in a data structure under the respective entity. Besides that, Algorithm 8 also covers the determination of the net execution times from the aforementioned timestamps. In order to comprehend this part of the algorithm, the intermediate states of the most relevant variables for determining the execution times of $Runnable_2$ are stated in the following:

$GET[Runnable_2, 0] = 14400 - 1000 = 13400$
$GET[Runnable_2, 1] = 24200 - 16000 = 8200$
$NET[Runnable_2, 0] = GET[Runnable_2, 0] = 13400$
$NET[Runnable_2, 1] = GET[Runnable_2, 1] = 8200$
$length(suspends[Runnable_2]) = 1)$
$preemption = resumes[Runnable_2] - suspends[Runnable_2] = 10400 - 5200 = 5200$
$NET[Runnable_1, 0] = NET[Runnable_1][0] - preemption = 13400 - 5200 = 8200$

The data structure that stores the net execution times is then used as input for Algorithm 9 which yields, for example, a constant value as the execution time distribution of $Runnable_2$:

$values = data[Runnable_2] = (8200, 8200)$
$length(values) = 2$
$value = mostFrequentValue(values) = 8200)$
$constant = 8200$
$minSL = kolmogorovSmirnovTest(8200, (8200, 8200)) = \sup \mid 8200 - (8200, 8200) \mid = 0$

6.3.5. Call Graph

As shown before, the runtime behaviour of a real-time system varies during execution. This is not only because of preemptions resulting from scheduling or because of hardware effects such as jitters but also as a consequence of the different execution sequences of a process. For example, a process calls a different runnble during

initialisation or in case of an error.

This manner, in which processes show a distinct behaviour in specific situations, is called mode dependency and is supported by both meta-models AUTOSAR and AMALTHEA. In contrast to AUTOSAR, whose mode management only allows one to define a mode-dependent execution of runnables, AMALTHEA allows one also to describe a probabilistic execution with the help of its call graph (see Chapter 4.2.1). Before such a graph can be constructed, all sequences of calls that are performed by a process are determined from a trace recording as described by Algorithm 10.

The preparation of the call graph is split in two different call sequence determinations. On the one hand, all runnable calls within a process are collected (see line 8 ff. in Algorithm 10), which constitute the content of the call graph. On the other hand, also the data accesses that are performed by a runnable are stored (see line 12 ff.). These are required to enable the detection of data dependencies during the composition of the call graph. Because data signals are accessed within the context of a process in BTF, the runnable instance that is currently executed is determined from the call graph first. The result is a three-dimensional matrix, in which each row represents a process or a runnable entity, each column an instance of that entity and the third dimension the sequence of calls performed by that instance.

The next step is to summarise the call sequences collected in the three-dimensional

Algorithm 10 Preparation of Call Graph

Input: trace – Tuple that contains all events in chronological order
Output: callSequences – Three-dimensional matrix that contains the observed calls for each process and runnable instance

```
1:  function prepareCallSequences(trace)
2:      callSequences ← [,,]
3:      for all event in trace do
4:          process ← event[source]
5:          pi ← event[sourceInstance]
6:          e ← event[entity]
7:          if event[type] = 'R' then                    ▷ Add runnable to call sequence
8:              if event[action] = 'start' then
9:                  i ← length(callSequences[process, pi)
10:                 callSequences[process, pi, i + 1] ← e
11:         else if event[type] = 'SIG' then             ▷ Associate signal access with runnable
12:             if event[action] = 'read' or 'write' then
13:                 i ← length(callSequences[process, pi])
14:                 [runnable, ri] ← callSequences[process, pi, i]
15:                 j ← length(callSequences[runnable, ri])
16:                 callSequences[runnable, ri, j + 1] ← e
17:     return callSequences
18: end function
```

Algorithm 11 Determination of Call Graph

Input: callSequences – Three-dimensional matrix that contains the observed calls for each process and runnable instance
processes – Set that contains the names of all process entities
Output: callGraph – Set of tuples representing the nested call sequences with their probabilities

```
1:  function determineCallGraph(callSequences, processes)
2:      callGraph ← ∅
3:      frequencies ← ()
4:      for all process ∈ processes do
5:          roots ← ∅
6:          for all sequence ∈ callSequences[processes] do
7:              frequencies[sequence] ← frequencies[sequence] + 1        ▷ absolute frequencies
8:              add sequence to patriciaTrie                             ▷ construct Patricia trie
9:              add sequence[1] to roots                                 ▷ collect graph roots
10:         for all root ∈ roots do
11:             amount ← cardinality(callSequences[process])
12:             branch ← createBranch(amount, root, patriciaTrie, frequencies)
13:             add branch to callGraph
14:     return callGraph
15: end function
```

matrix by building a call graph, which is described in Algorithm 11. There, redundant information such as storing the same call sequence multiple times is eliminated by determining the amount of times each unique call sequence was observed (see line 6f). To further reduce redundancy and to achieve a more compact call graph, not only unique sequences are considered but also similar ones. For that reason, all observed call sequences are also used to build up a PATRICIA Trie (see line 8), which is an optimised data structure for retrieving strings with a common prefix. Thus, each call in a call sequence is equivalent to a character in a string.

These two pieces of information the frequency of the unique call sequences and the PATRICIA Trie, are then used to build up the call graph, resp. its branches in a recursive manner as described in Algorithm 12. Starting with the roots, i.e., the first element stored in a the call sequences and, thus, the first calls performed by a process, one call after another is added to the call graph. If a common sequence of calls is succeeded by two different calls, a fork in the graph is created and the probability for taking a specific branch is calculated. To do so, all sequences that start with the same calls, the prefix, are retrieved from the PATRICIA Trie (see line 4). Then, the probability is determined by the frequency that each of these retrieved call sequences was observed (see line 6 ff.). This is continued recursively by adding the next succeeding call to the prefix (see line 21) until the PATRICIA Trie yields only one sequence.

Algorithm 12 Create Branch

Input: amount – Integer value stating the amount of call sequences covered by the parent branch
prefix – String that contains an observed sequence of calls
patriciaTrie – Patricia Trie containing all observed sequences of calls
frequencies – Tuple of integers containing how often sequences of calls were observed
Output: callGraph – Set of tuples representing the nested call sequences with their probabilities

```
1: function createBranch(amount, prefix, patriciaTrie, frequencies)
2:     callGraph ← ∅
3:     sequences ← ∅
4:     sequences ← prefix(patriciaTrie, prefix)          ▷ get sequences with matching prefix
5:     frequency ← 0
6:     for all sequence ∈ sequences do
7:         frequency ← frequency + frequencies[sequence]          ▷ how often did a sequence occur
8:     probability ← frequency / amount
9:     if cardinality(sequences) = 1 then
10:        n ← length(prefix)−1
11:        m ← length(sequences[0])−1
12:        callSequence ← sequences[0, n ... m]
13:        branch ← (probability, callSequence)          ▷ no more recursion
14:        add branch to callGraph
15:    else
16:        subBranches ← ∅
17:        for all sequence ∈ sequences do
18:            n ← length(prefix)−1
19:            m ← length(prefix)
20:            newPrefix ← sequence[0 ... m + 1]
21:            subBranch ← createBranch(frequency, newPrefix, patriciaTrie, frequencies)
22:            add subBranch to subBranches
23:        callSequence ← (sequence[n ... m], subBranches)
24:        branch ← (probability, callSequence)
25:        add branch to callGraph
26:        checkModeDependency(callGraph)
27:        minimise(callGraph)
28:    return callGraph
29: end function
```

After all branches have been created, it is possible to check whether a mode dependency is the cause that leads to different call sequences (see line 26 in Algorithm 12). As the name already implies, mode dependent execution varies due to a dedicated state of a variable and, thus, the first action that has to be performed during execution is a check of that variable. As a consequence, this function looks into each branch of the determined call graph and checks if the first action performed by the called runnable is a data access to the same variable. If that is the fact and if the values of that variable are mutually exclusive for each branch at the time of the data access, a data dependency can be concluded.

Finally, the call graph is examined and if possible further minimised. To do so,

common calls at the end of the call sequences are determined with the help of a suffix tree correspondingly to how the PATRICIA Trie is used and combined in a single sequence of calls.

Example 5 (Call Graph). The creation of a call graph is shown with the help of the simple system that was introduced first in Example 1. Listing 6.6 depicts a BTF trace that covers 58 ms of the system's internal behaviour. The intermediate states of the most relevant variables used in Algorithm 10 to gather the information for creating a call graph later on via Algorithm 11 are added in red.

```
0,CORE0,0,T,Task_1,0,activate
100,CORE0,0,T,Task_1,0,start
100,Task_1,0,R,Runnable_1,0,start
  callSerquences[Task_1,0] = (Runnable_1)
500,CORE1,0,T,Task_3,0,activate
600,CORE1,0,T,Task_3,0,start
1000,Task_1,0,R,Runnable_1,0,terminate
1000,Task_1,0,R,Runnable_2,0,start
  callSerquences[Task_1,0] = (Runnable_1,Runnable_2)
5000,CORE0,0,T,Task_2,0,activate
5200,Task_1,0,R,Runnable_2,0,suspend
5200,CORE0,0,T,Task_1,0,preempt
5200,CORE0,0,T,Task_2,0,start
10200,CORE0,0,T,Task_2,0,terminate
10400,CORE0,0,T,Task_1,0,resume
10400,Task_1,0,R,Runnable_2,0,resume
14400,Task_1,0,R,Runnable_2,0,terminate
14400,CORE0,0,T,Task_1,0,terminate
15000,CORE0,0,T,Task_1,1,activate
15100,CORE0,0,T,Task_1,1,start
15100,Task_1,1,R,Runnable_1,1,start
  callSerquences[Task_1,1] = (Runnable_1)
16000,Task_1,1,R,Runnable_1,1,terminate
16000,Task_1,1,R,Runnable_2,1,start
  callSerquences[Task_1,1] = (Runnable_1,Runnable_2)
24200,Task_1,1,R,Runnable_2,1,terminate
24200,CORE0,0,T,Task_1,1,terminate
```

```
28 30000,CORE0,0,T,Task_1,2,activate
   30100,CORE0,0,T,Task_1,2,start
30 30100,Task_1,2,R,Runnable_1,2,start
     callSerquences[Task_1,2] = (Runnable_1)
32 31000,Task_1,2,R,Runnable_1,2,terminate
   31000,Task_1,2,R,Runnable_4,0,start
34   callSerquences[Task_1,2] = (Runnable_1,Runnable_4)
   38000,Task_1,2,R,Runnable_4,0,terminate
36 38000,CORE0,0,T,Task_1,2,terminate
   45000,CORE0,0,T,Task_1,3,activate
38 45100,CORE0,0,T,Task_1,3,start
   45100,Task_1,3,R,Runnable_3,0,start
40   callSerquences[Task_1,3] = (Runnable_3)
   50100,Task_1,3,R,Runnable_3,0,terminate
42 50100,Task_1,3,R,Runnable_4,1,start
     callSerquences[Task_1,3] = (Runnable_3,Runnable_4)
44 57100,Task_1,3,R,Runnable_4,1,terminate
   57100,CORE0,0,T,Task_1,3,terminate
46 58000,CORE1,0,T,Task_3,0,terminate
```

Listing 6.6: Example trace for showing CoreTAna's algorithms for determining the call graphs

Listing 6.4 contains only the runnables that are called by $Task_1$. Thus, these are the only pieces of information that are gathered for determining a call graph. To do so, every runnable entity that is called within the context of that task is stored in a data structure under the respective entity and instance. This data structure is then used as input for Algorithm 10 which yields together with Algorithm 12 the call graphs. In order to comprehend these two algorithms, the intermediate states of the most relevant variables for determining the call graph of $Task_1$ are stated in the following:

$frequencies[(Runnable_1, Runnable_2)] = 2$
$frequencies[(Runnable_1, Runnable_4)] = 1$
$frequencies[(Runnable_3, Runnable_4)] = 1$
$roots = \{Runnable_1, Runnable_3\}$
$amount = 4$

One branch starting with the call of $Runnable_1$ is created by Algorithm 12 the follow-

ing way:

$sequences = \{(Runnable_1, Runnable_2), (Runnable_1, Runnable_4)\}$
$frequency = 2 + 1 = 3$
$probability = \frac{frequency}{amount} = \frac{3}{4}$
$cardinality(sequences) = 2$
$sequence = (Runnable_1, Runnable_2)$
$n = length((Runnable_1)) - 1 = 0$
$m = length((Runnable_1)) = 1$
$newPrefix = (Runnable_1, Runnable_2)$

Then, the first recursion happens:

$sequences = \{(Runnable_1, Runnable_2)\}$
$frequency = 2$
$probability = \frac{frequency}{amount} = \frac{2}{3}$
$cardinality(sequences) = 1$
$n = length((Runnable_1, Runnable_2)) - 1 = 1$
$m = length((Runnable_1, Runnable_2)) - 1 = 1$
$callSequence = sequences[0, 1..1] = (Runnable_2)$
$branch = (\frac{2}{3}, (Runnable_2))$
$callGraph = \{(\frac{2}{3}, (Runnable_2))\}$

The recursion ends and returns to the calling function:

$subBranch = (\frac{2}{3}, (Runnable_2))$
$sequence = (Runnable_1, Runnable_4)$
$subBranch = (\frac{1}{3}, (Runnable_4))$
$subBranches = \{(\frac{2}{3}, (Runnable_2)), (\frac{1}{3}, (Runnable_4))\}$
$callSequence = (Runnable_1, \{(\frac{2}{3}, (Runnable_2)), (\frac{1}{3}, (Runnable_4))\})$
$branch = (\frac{3}{4}, (Runnable_1, \{(\frac{2}{3}, (Runnable_2)), (\frac{1}{3}, (Runnable_4))\}))$

Finally, the algorithm yields the following call graph for $Task_1$ which says that in 75 % of the cases $Runnable_1$ is called first followed by $Runnable_2$ in $66.\overline{6}$ % of the cases and $Runnable_2$ in the other $33.\overline{3}$ % of the cases and $Runnable_3$ followed by $Runnable_4$ is called with a probability of 25 %:

$$callGraph = \{(\frac{3}{4}, (Runnable_1, \{(\frac{2}{3}, (Runnable_2)), (\frac{1}{3}, (Runnable_4))\})), (\frac{1}{4}, (Runnable_3, Runnable_4))\}.$$

6.4. Summary

The algorithms presented in this chapter show that it is possible to determine key characteristics of a system from a trace recording. More importantly, a lot of information can be extracted directly from a trace recording without the need of much inference. This makes CoreTAna's behaviour comprehensible and reproducible.

Structuring the algorithms in three consecutive steps highlights the fact that there are only few dependencies between the determination of individual system characteristics. In fact, all algorithms within a step can run in parallel, and, CoreTAna can achieve results in a reasonable amount of time even from trace recordings that span multiple gigabytes. It is also shown how the performance of CoreTAna's algorithms is further optimised by storing the trace in a database, and then querying just the information required for the individual algorithm instead of chronologically processing each event.

How well our algorithms reversely engineer a system's timing behaviour and how long it takes CoreTAna to process a trace recording are topics that are addressed next.

7

Distance of Timed Actions

Before evaluating the efficiency of CoreTAna's algorithms, we first discuss the challenge of evaluating how well a synthesised model reflects the timing behaviour of the original system.

Because CoreTAna performs a dynamic analysis, the only artefact that can be used for the evaluation of CoreTAna's quality of reverse engineering are the events that occur during system execution and the specific moments in time at which they are observed. As the synthesised model ideally reflects the same temporal behaviour,

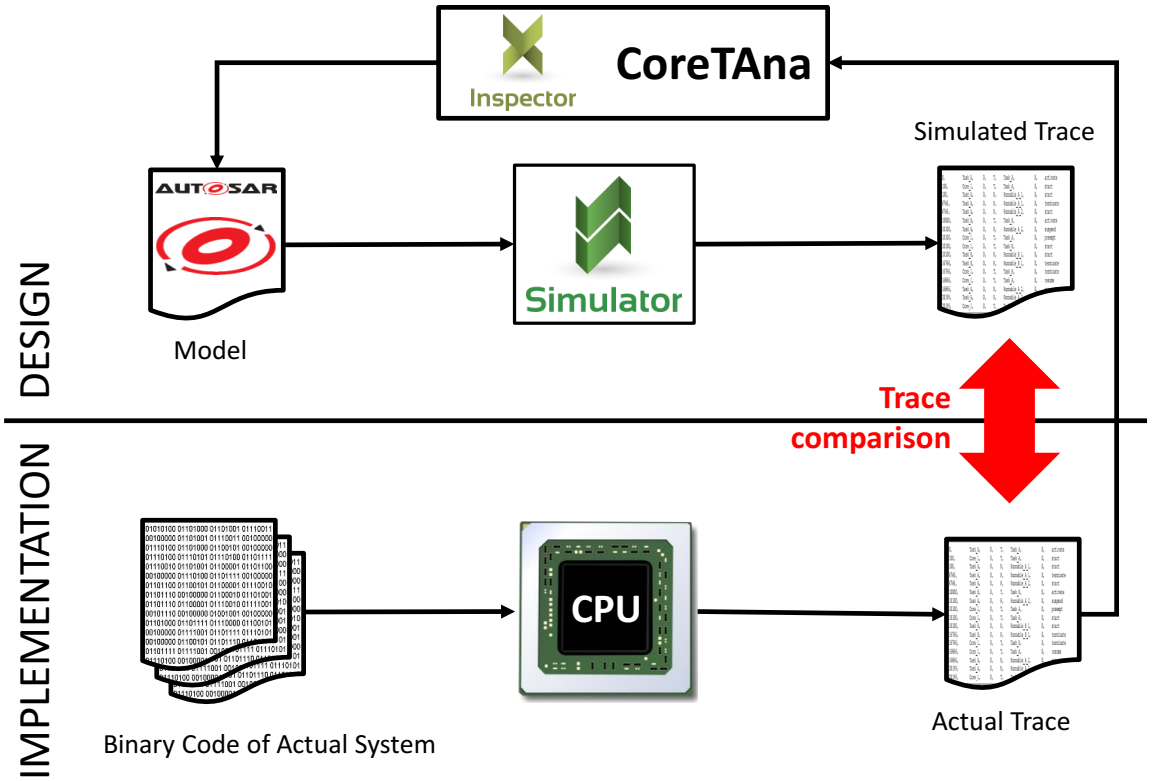


Figure 7.1.: Trace Comparison. Schematic approach for analysing how closely an AUTOSAR model that has been synthesised by CoreTAna reflects the actual system. Reprinted from [13].

the most obvious approach to evaluate a reverse engineering solution is to compare the trace recordings of the actual system with those generated by simulating the synthesised model. As depicted in Fig. 7.1, CoreTAna utilises the TA Simulator [70] for this purpose. This commercial model-based tool, which is used by many Tier-1s and OEMs in the automotive industry, allows one to simulate the timing behaviour of models of AUTOSAR-compliant real-time systems.

As presented in Chapter 7.2, a lot of research has been conducted regarding the comparison of trace recordings. Most focuses on statistical methods, which yield, in our case, results that suggest an insufficient correspondence. This is due to the fact that a model represents just an abstraction of a real system, where characteristics are summarised at the level of detail that is defined by the meta-model. Thus, a model describes rather a similar system than the exact same system that underlies the model. However, statistical methods look for an identical behaviour, which cannot be provided because of the abstraction.

Because of this absence of a suitable way to assess the quality of reverse engineered real-time software, we have developed a new measure. Before we go into details about the definition of our measure, we first present the challenges that have to be met when evaluating differences in the timing behaviour of real-time systems.

7.1. Challenges of Comparing Real-time Behaviour

Each alteration of a system's characteristic, such as a task's priority, can have an impact on the timing behaviour of the entire system. For that reason, we present in the following a set of models that represent common architectural patterns in the real-time software domain and feasible variations for each pattern. The goal of these variations is to provoke realistic changes to a system's timing behaviour. The challenge of a suitable measure is then to quantify the impact of such a variation in a reasonable way.

The basic idea for this approach is inherited from the work of Huselius [9, p. 111ff], where so-called Archetypes, which represent the architectural patterns, and feasible variations for each pattern, so-called PICs, are described. Unfortunately, Huselius only gives a general description of each Archetype, which makes it impossible for us to reproduce them precisely. Nevertheless, our models cover all Archetypes originally

introduced by Huselius, but have been extended by additional variations to consider AUTOSAR-specific aspects. To highlight the impact of a system change made by a variation, the variations are designed in a consecutive way such that each variation within a pattern alters the previous one by a single aspect. All systems that are described in this section are public as example models in the Eclipse APP4MC Release Version 0.7.2[1], which is an open-source platform for engineering embedded multi- and many-core software systems.

7.1.1. Purely Periodic without Communication

This system architecture pattern (see Appendix A.1.1) consists of seven tasks, where each task is activated periodically and no data accesses are performed. The execution time for each task is determined by so-called *runnable entities*. All tasks contain just one runnable, except for T_7 which calls at first $R_{7,1}$ and then $R_{7,2}$. The variations applied to this system pattern are:

1) Initial Task Set: Tasks T_4, T_5, T_6, and T_7 are active and scheduled according to fixed-priority preemptive scheduling.

2) Increase of Task Set Size I: Tasks T_3, T_4, T_5, T_6, and T_7 are active. Hence, system utilisation is further increased.

3) Increase of Task Set Size II: Tasks T_1, T_3, T_4, T_5, T_6, and T_7 are active, i.e., system utilisation is further increased.

4) Increase of Task Set Size III: From this variation through to variation 8, all tasks ($T_1 - T_7$) are active. This increases system utilisation again.

5) Schedule: From this variation through to variation 8, T_7 is set to non-preemptive. Hence, the system's timing behaviour is changed, which results in extinct activations.

6) Activation: From this variation through to variation 8, the maximum number of queued activation requests is set to 2 for all tasks. This solves the problem with extinct activation request that result from queue overflows in the previous variation.

7) Schedule Point: A scheduler call is added to T_7 between the calls of $R_{7,1}$ and $R_{7,2}$. This changes the timing behaviour.

8) Scheduling Algorithm: The scheduling algorithm is set to *Earliest Deadline First*, so that the timing behaviour is changed completely.

[1] `https://www.eclipse.org/app4mc/`

7.1.2. Client-Server without Reply

This system architecture pattern (see Appendix A.1.2) extends the previous one by adding one-way communication between tasks. Task T_1 sends periodically a message to task T_2. Each time, one of the five available contents of the message is chosen according a defined probability distribution. For example, in 15 % of the cases the message contains the value "0". Task T_2 reacts on the received message by showing a distinct behaviour for each content, which is manifested by varying execution times. The implemented task set is depicted in Fig. 7.2.

The variations applied to this system pattern are:

1) Initial Task Set: All tasks as defined above are scheduled according to fixed-priority preemptive scheduling.

2) Exclusive Area: For this variation, all data accesses are protected by a mutex and priority ceiling protocol. Hence, blocking situations appear.

3) Inter-Process Activation: As from this variation on, task T_2 gets activated by an inter-process activation from task T_1, so that a direct connection between T_1 and T_2 is established.

4) Priority Ordering: As from this variation on, the priority relation between tasks

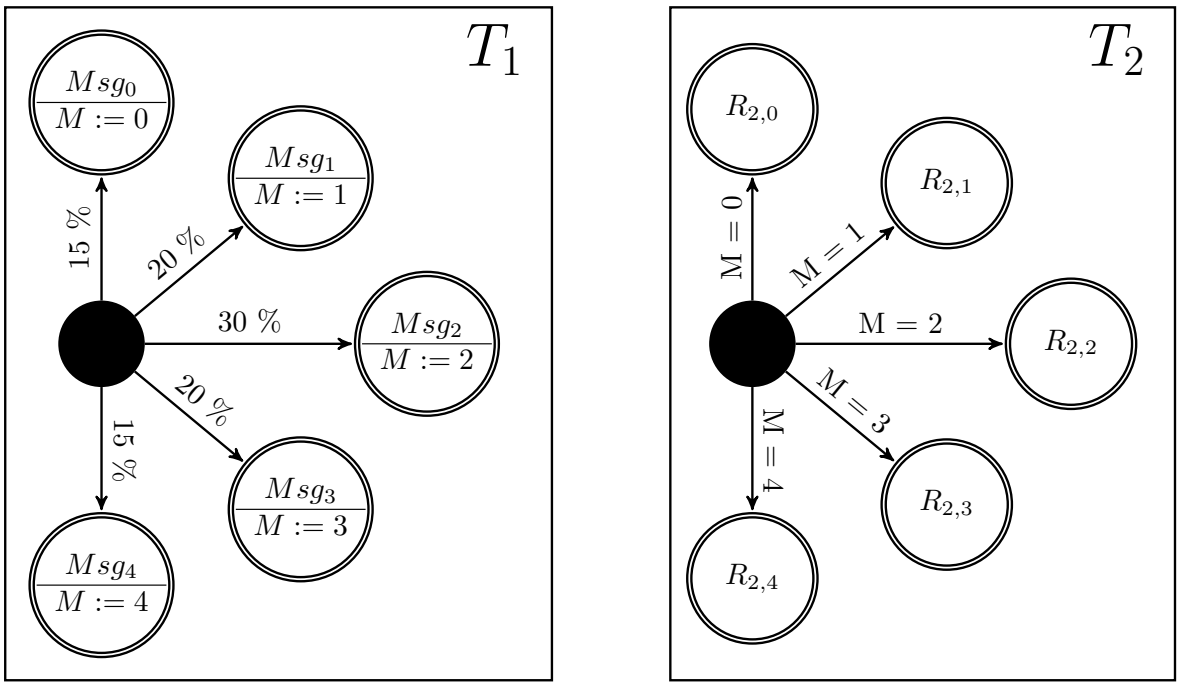


Figure 7.2.: Client-Server without Reply. State diagram implemented by the system architecture pattern 'Client-Server without Reply'. Reprinted from [13].

T_1 and T_2 is reversed. Thereby, a switch from asynchronous to synchronous communication is realised.

5) **Event Frequency Increase:** As from this variation on, the periodicity of T_1 is shortened so that system utilisation is increased.

6) **Execution Time Fluctuation:** As from this variation on, the execution time distribution is widened for both tasks. Hence, system utilisation is increased further, which results in extinct activations.

7) **Activation:** As from this variation on, the maximum number of queued activation requests for both tasks is set to 2. Thereby, the problem with extinct activations resulting from the previous variation is solved.

7.1.3. State Machine

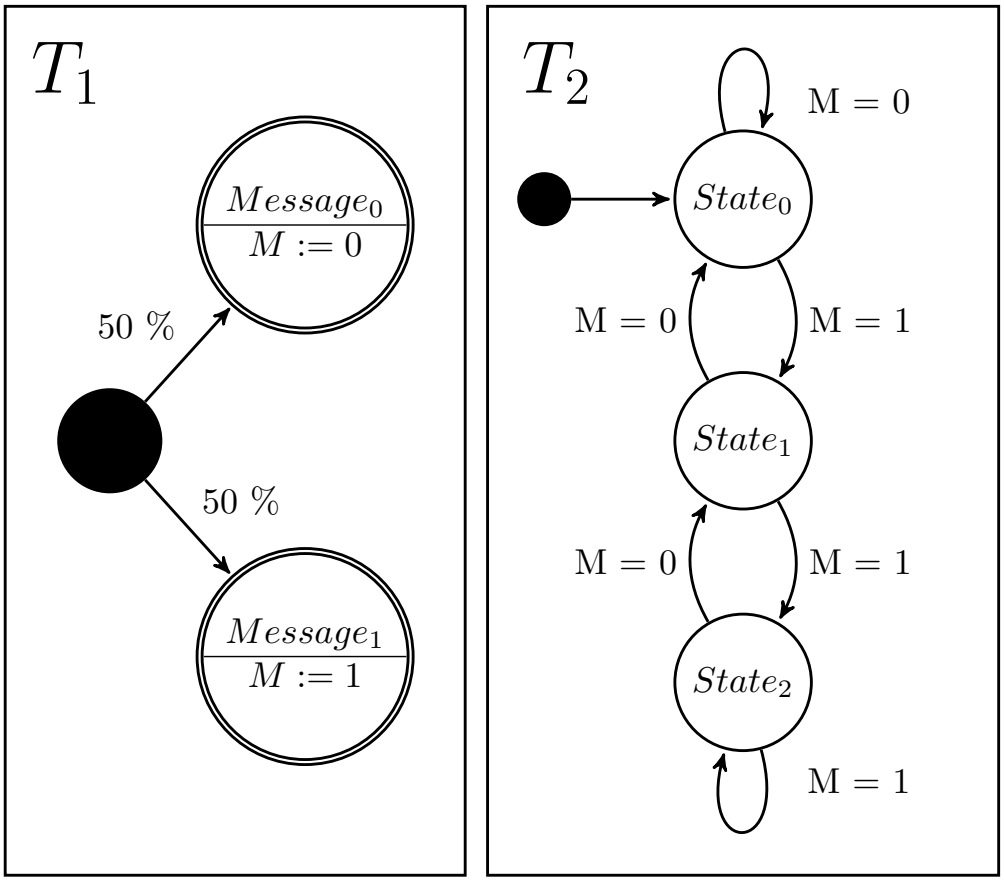


Figure 7.3.: State Machine. State diagram implemented by the system architecture pattern 'State Machine'. Reprinted from [13].

In this system architecture pattern (see Appendix A.1.3), the previous one is extended in such a way that the task T_2 that receives messages varies its dynamic behaviour,

and consequently its execution time not only according to the transmitted content, but also according to its current internal state, i.e., the previously transmitted contents.

Task T_1 sends periodically a message to task T_2. With a probability of 50 %, the message contains the value "1". In the rest of the cases the content is "0". Initially starting from state $State_0$, task T_2 transitions to the next state, if the received message contains the value "1". Otherwise, it returns to the state it has been in before. Each state manifests in a different execution time, which represents the varying dynamic behaviour. The implemented state machine is depicted in Fig. 7.3.

The variations applied to this system pattern are equal to those described in Sec. 7.1.2. Only the sequence in which the underlying system is modified by each variation is slightly changed, in order to provoke realistic challenges such as exceeding the maximum number of queued activation requests.

7.1.4. Feedback Loop

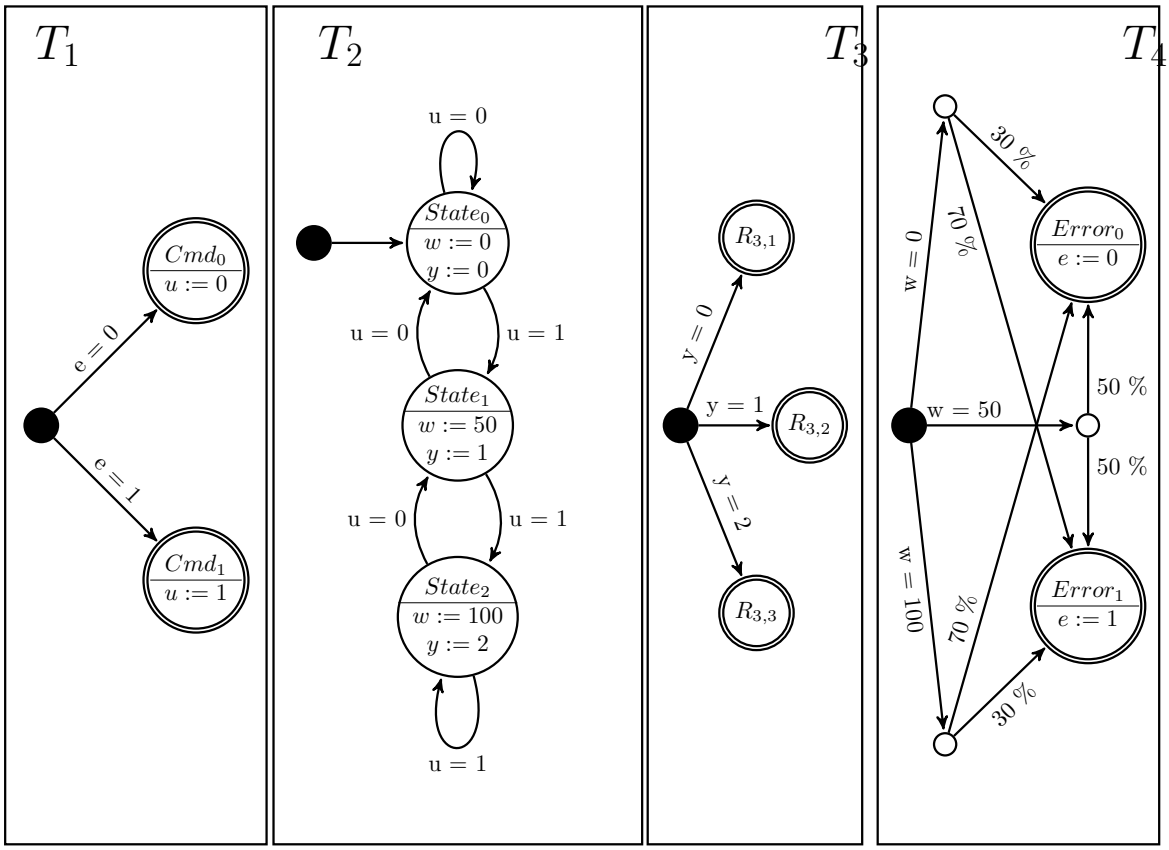


Figure 7.4.: Feedback Loop. State diagram implemented by the system architecture pattern 'Feedback Loop'. Reprinted from [13].

The task set of the previous system architecture pattern is expanded further so that messages are exchanged in a loop, instead of just in one way (see Appendix A.1.4). Each task represents a part of a feedback control system. Task T_1 represents the controller, which gets the measured error e as input and sets the system input u accordingly. The system, which is defined by task T_2, implements a state machine that is triggered by the system input. Depending on the state the task is in, it produces the measured output w and the system output y. The latter influences the environment, which is defined by task T_3, and manifests in varying execution times. Finally, the feedback loop is implemented by task T_4, which sets the measured error e according the measured output of the system w and a given probability.

In addition to the feedback loop as depicted in Fig. 7.4, other system architecture patterns are added to be executed concurrently, in order to increase the complexity. Tasks T_5 and T_6 represent a client-server without reply, and task T_7 is a periodically activated task without any communication. The variations for this system pattern are equal to those applied to the previous patterns described in Secs. 7.1.2 & 7.3. However, the changed characteristics of this task set in comparison to the previous ones required again to slightly change the sequence in which the variations are applied.

7.1.5. State Machine Feedback Loop

Finally, the previous system architecture pattern is expanded further by combining the ideas behind patterns *State Machine* and *Feedback Loop* (see Appendix A.1.5) This means that messages are exchanged in a loop, and each sender/receiver is also a state machine. In addition to the state machine feedback loop as depicted in Fig. 7.5, other system architecture patterns are again added to be executed concurrently, to increase complexity. Tasks T_3 and T_4 represent a client-server without reply, and task T_5 is a periodically activated task without any communication.

The variations and the sequence in which these are applied to this system pattern are identical to those used for pattern ‘Feedback Loop’ in Sec. 7.1.4.

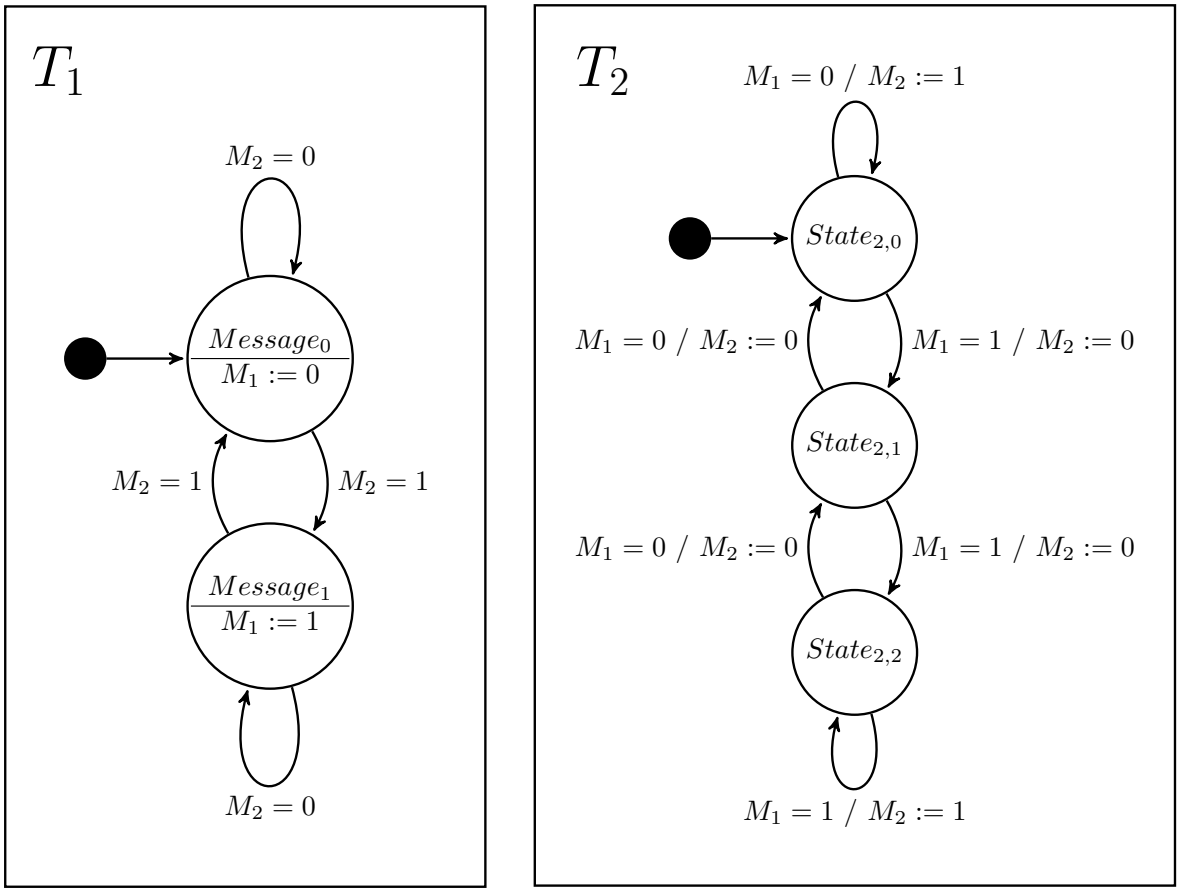


Figure 7.5.: State Machine Feedback Loop. State diagram implemented by the system architecture pattern 'State Machine Feedback Loop'. Reprinted from [13].

7.2. Related Work

The problem of comparing trace recordings from embedded real-time systems regarding their temporal behaviour is similar to the problem of simulation model validation [73]. Therefore, many different techniques are available that tackle the problem of simulation model validation [74–76]. The following two objective techniques can mainly be found in the domain of real-time systems. One is based on statistical methods such as goodness-of-fit tests [4, 73], and the other technique uses Algebra.

Lu et. al [73] study different ways to identify temporal differences between real-time systems. To do so, the authors check with statistical hypothesis testing, if the execution times (ETs) of tasks, resp., their response times (RTs), which are determined from timing traces, are in each case from the same population. A possible use of parametric tests, e.g., t-test, z-test, etc. is ruled out first, because tests have shown that sampling distributions of ETs and RTs do not conform to any known distribution like

uniform or normal distribution. The resampling methods bootstrap and permutation test are also not employable, since they "failed in, for instance, identifying temporal differences" [73, p. 289]. Finally, the Chi-squared Test (χ^2 Test), K-S Test, and Wilcoxon–Mann–Whitney test are examined as representatives for non-parametric statistical hypothesis tests. The former was found to be not suitable for timing analysis, because the required a priori knowledge is either too limited or too subjective. In case of the Wilcoxon–Mann–Whitney test the authors have argued that this test may come to an incorrect conclusion when considering multi-model distributions, which are realistic in the attended context thought. Finally, the K-S Test has been evaluated using a fictive system, since this non-parametric statistical hypothesis test does not feature the aforementioned drawbacks of the χ^2 Test and the Wilcoxon–Mann–Whitney test. Although this evaluation shows that the results are in line with the authors expectations, their algorithm can only detect significant differences in the ETs and RTs of tasks. Further, variations that do not impact the ET or RT of tasks, like drifting activation instants, are not considered at all.

Other authors [77] argue against the use of statistical techniques for comparing trace recordings. Because the χ^2 Test of independence is categorical in the temporal dimension, it "leads to unintuitive results due to false negatives" [77, p. 4]. The execution time distribution of a program is often very complex and cannot be fitted to a known distribution. Because the "Kolmogorov-Smirnov test [...] assumes that one of the distributions in a comparison is mathematically modeled" [77, p. 4], this statistical test cannot be taken into consideration for comparing metrics determined from trace recordings.

Because of the aforementioned unfitness of statistical methods in our case and the consequent fact that the measure that we introduce in the next section is based on Algebra, we present in the following related work that focuses on algebraic procedures.

Miranskyy et al. [78] propose a more generic approach by using Shannon and extended entropy measures for comparing traces. A trace can be represented as a string, in which each trace record is encoded by a unique character. Based on this string, the probability of events are extracted and the entropy measures are calculated. With the help of the proposed measure, multiple entropy-based fingerprints for a trace are calculated that allows one to quantify the distance between a pair of traces. However, the intended use case for this approach is to classify trace recordings, e.g., finding a trace that shows defects of a known bug rather then defining a distance between each pair.

Furthermore, the trace recording that are considered have no notion of time, which is essential in our case.

Anderson discusses in [28, p. 108 ff.] the equivalence of observable properties such as response-times, patterns, and resource utilizations in trace recordings. Nevertheless, no explicit tolerances are suggested for these comparison properties.

Huselius introduces in [9, p. 143 ff.] an objective measurement called *Sum of Divergence (SoD)*, which establishes a bijective mapping between two response time samples in order to summarise the differences in their distributions. Disadvantages of this solution are, on the one hand that the least common multiple (LCM) is used to obtain equally sized sampled distributions. On the other hand, the measure is normalised by the maximum difference between samples in the distributions. Thus, a single scattered outlier can have a big impact on the quality of the result, because the outlier can, e.g., be multiplied by the LCM or result in an improper difference for normalisation.

Nemati et al. describe in [79] an algorithm that first divides trace recordings into equally sized time windows and then calculates the differences in resource consumption properties, such as the execution times of tasks between correlating time windows. However, not only is it difficult to determine a leeway for difference in advance, but the traces also have to start from exactly the same state and have to contain the same sequences of states. The latter cannot be guaranteed for probabilistic models.

Because of the absence of a suitable way to assess the quality of reverse engineered real-time software, we introduce a new measure.

7.3. Definition

The goal of our measure is to quantify the difference between two traces regarding the recorded timing behaviour. As described in Chapter 5.3.3, each trace contains the occurrences of events that are observed during system execution. Because an event is defined as an action that is observed in the system at a specific point in time, we call our measure DoTA. DoTA is hierarchically structured in two steps: *Amount Distance* and *Entity Distance*. The former step checks whether both trace recordings contain the same entities and the latter compares the temporal behaviour of each entity.

The BTF shows, that actions are performed by a multitude of named elements in the system, the so-called entities, such as the tasks, ISRs, and runnables. For that reason, our measure checks, at first, by calculating the *Amount Distance* whether all

entities in the one trace are also present in the other trace.

Definition 7.1. Amount Distance Δ_A: Let $P(s_i)$ be the process entities in trace sample s_i for $i \in 1,2$, and let $|X|$ denote the cardinality of a set X. The Amount Distance Δ_A is defined by

$$\Delta_A(s_1, s_2) = 1 - \frac{|P(s_1) \cap P(s_2)|}{|P(s_1) \cup P(s_2)|}. \tag{7.1}$$

To compare the amount of identical entities in two traces, Eq. 7.1 considers just process entities, i.e., the tasks and ISRs, and not all observable entities. This is for multiple reasons. On the one hand, the temporal behaviour of a real-time system is primarily determined by the processes. On the other hand, technical limitations rule out the recording of all entities at once, whereas a reversely engineered model can contain additional entities in order to reproduce correct behaviour. Nevertheless, the temporal behaviour of processes in both samples have to match, which then indirectly reflects also a correct representation of other entities, e.g., runnables.

In a second step, the *Entity Distance* uses the metrics that we introduced in Chapter 5.3.5 and that allow one to objectively analyse a system's temporal behaviour to determine the correspondence between observed entities in the comparative trace recordings.

Definition 7.2. Entity Distance Δ_E: Let $E(s_i) \subseteq E(s_i)$ be all task, ISR, and runnable entities in a trace sample s_i for $i \in 1,2$ and such that $|E(s_1) \cap E(s_2)| > 0$. The Entity Distance Δ_E is defined by

$$\Delta_E(s_1, s_2) = \sum_{e \in E(s_1) \cap E(s_2)} \frac{\sqrt{\sum_{m \in M} \frac{1}{|M|} \cdot (m_{s_1} - m_{s_2})^2}}{|E(s_1) \cap E(s_2)|}, \tag{7.2}$$

where $M = \{min_t, max_t, \bar{x}_t, Q_{1,t}, Q_{2,t}, Q_{3,t}, IQM_t \mid \forall t \in \{\text{NET, A2A, SD, Ready, Parking, Polling, Waiting}\}\}$; here, min_t is the minimum, max_t the maximum, $\bar{x}_t$ the mean, $Q_{1,t}$ the lower quartile, $Q_{2,t}$ the median, $Q_{3,t}$ the upper quartile, and IQM_t the interquartile mean of a time metric t.

The Entity Distance Δ_E determines the differences between all the entities, including runnables, tasks and ISRs, that are available in both samples. These entities are compared by calculating the weighted Euclidean distance between the temporal behaviour recorded in the one sample and that in the other sample. Due to the fact that each trace recording typically contains multiple observations of the same entity and,

thus, also a variety of temporal characteristics, measures of descriptive statistics such as the minimum, maximum, mean, lower quartile, median, upper quartile, and the inter-quartile mean are used to quantitatively summarise the general behaviour for each entity. Measures of spread or shape are not considered because they would lead to inconsistent results, e.g., if the variances of two comparable entities are exactly the same but their individual times are far apart. In this case, the temporal behaviour is completely different, but an alignment in the variance would presume otherwise. Instead, multiple measures of location are employed by us to capture differences in variability. The weight of the Euclidean distance is chosen in such a way that each metric measure contributes to the same extent, because none outranks the others in importance.

Definition 7.3. Distance of Timed Actions (DoTA): Let s_1, s_2 be two sample trace recordings, let Δ_A denote the Amount Distance, and let Δ_E be the Entity Distance. Then, the Distance Timed Actions (DoTA) is defined by

$$DoTA(s_1, s_2) = 1 - [1 - \Delta_A(s_1, s_2)] \cdot [1 - \Delta_E(s_1, s_2)]. \tag{7.3}$$

Finally, the equality in the amount of same entities in both samples and the differences of each individual entity are combined in our measure *Distance of Timed Actions (DoTA)*, in order to obtain the distance between two sample trace recordings.

7.3.1. Example

In the following, the usage of the previously introduced distance measure is shown on the basis of a simple example. Let us consider the two sample trace recordings shown in Listings 7.1 and 7.2, where a single task T gets executed in each one:

```
0, T, activate
0, T, start
5, T, terminate
10, T, activate
10, T, start
16, T, terminate
```

Listing 7.1: Trace Sample 1

```
1, T, activate
1, T, start
5, T, terminate
12, T, activate
12, T, start
17, T, terminate
```

Listing 7.2: Trace Sample 2

Since the same task occurs in both samples, the Amount Distance Δ_A is calculated as follows using Eq. 7.1:

$$\Delta_A(s_1, s_2) = 1 - \frac{|\{T\} \cap \{T\})|}{|\{T\} \cup \{T\}|} = 1 - \frac{1}{1} = 0$$

Before the Entity Distance Δ_E can be determined, the real-time metrics t for each task occurrence have to be calculated first. In a next step, these metrics are scaled to bring all values and, consequently, also the final distance result into the range [0,1]. For this purpose, each metric value is scaled to the maximum value for the metric in both samples: $x'_t = \frac{x_t}{max_t(s_1,s_2)}$. Finally, the scaled values are summarised via the aforementioned measures of descriptive statistics. For simplicity, we consider here just the mean $\bar{x}'_t$ for each metric. The means resulting from these calculations are listed in Table 7.1.

Based on them, the Entity Distance Δ_E can then be calculated by applying Eq. 7.2. The weigh for each metric m is balanced and consequently set to $\frac{1}{6}$, because of the six metrics used ($\bar{x}'_{A2A}, \bar{x}'_{NET}, \bar{x}'_{Parking}, \bar{x}'_{Polling}, \bar{x}'_{Ready}, \bar{x}'_{SD}$).

$$\Delta_E(s_1, s_2) = \frac{\sqrt{\frac{1}{6} \cdot (0.\overline{90} - 1)^2 + \frac{1}{6} \cdot (0.91\overline{6} - 0.75)^2 + 0}}{|\{T\}|}$$
$$\approx 0.077$$

Hence, the sample traces s_1 and s_2 differ by roughly 7.7 % according to our measure DoTA.

7.3.2. Analysing Differences in Trace Recordings

The example that is given in Chapter 7.3.1 illustrates, how DoTA is applied on traces to quantify their differences. But how does this information help an engineer and, more importantly, how can an engineer figure out the reason of an unexpected result?

DoTA itself yields a single value that represents the difference between two trace recordings in percentage. A value close to zero means, that the timing behaviour recorded in the traces are basically equal. The higher the result gets and the closer it gets to 100 %, the further away are the traces and, thus, the systems' timing behaviours. In case of the example, the difference was 7.7 %. Because the Euclidean distance used in our measure has a quadratic characteristic, even small variations have an extensive

Table 7.1.: Metric Results of the Example. Real-time metrics t for task T in trace samples s_1 and s_2 and their scaled means $\bar{x}'_t$.

t	s_1			s_2		
	X_t	X'_t	$\bar{x}'_t$	X	X'_t	$\bar{x}'_t$
$A2A$	{10}	$\{\frac{10}{11}\}$	$0.\overline{90}$	{11}	$\{\frac{11}{11}\}$	1
NET	{5,6}	$\{\frac{5}{6}, \frac{6}{6}\}$	$0.91\overline{6}$	{4,5}	$\{\frac{4}{6}, \frac{5}{6}\}$	0.75
$Parking$	{0,0}	{0,0}	0	{0,0}	{0,0}	0
$Polling$	{0,0}	{0,0}	0	{0,0}	{0,0}	0
$Ready$	{0,0}	{0,0}	0	{0,0}	{0,0}	0
SD	{0,0}	{0,0}	0	{0,0}	{0,0}	0

impact on the result and get highlighted by DoTA.

DoTA gives an overview of the similarity of the entire systems and is, consequently, the place to start investigating from in case of indicated dissimilarity. Because the measure is hierarchically structured in two parts, a first step for an engineer is to look at the result of the *Amount Distance* to check, if all both trace recordings contain the same entities. This measure is designed in such a way that it can be interpreted in an easy way. For example, let us consider a system with five tasks and the second trace is missing a task for some reason, then the similarity of both traces is $\frac{4}{5} = 80\,\%$ at most, because four tasks are in both trace recordings and an additional one is only present in the one trace.

In a next step, the *Entity Distance* is considered. Because the result of this measure aggregates the differences of all entities in the system, this value gives just an overview and lacks the possibility to highlight the exact reason for dissimilarity. However, the big advantage of this measure is that it can be used for any set of entities not only for all system entities. This means, that an engineer can apply the *Entity Distance* to all processes and all runnables separately to identify whether the dissimilarity occurs at process level, resp., at system level. This is done in Chapter 8.2, where Fig. 8.6 contrasts the differences of all system system entities, tasks and runnables.

With this information, the search for the source of the indicated dissimilarity continues by applying the *Entity Distance* to individual entities. In case the result of the

distance measure at process level is conspicuous, the differences for each task and ISR are determined as it is done in Chapter 7.5.1. There, Fig. 7.8 shows the results of *Entity Distance* for all processes that are present in both trace recordings in a radar chart. That way, the contributing difference of each individual entity to the entire dissimilarity is highlighted which allows one to identify outliers in an easy way.

These actions are repeated with the individual metrics of the entity under investigation to see which metrics have changed for a specific entity between the two trace recordings and to what extend. Finally, if the metric of interest is know, the individual values of the entities metric are compared, e.g., by contrasting them in a histogram. Fig. 7.6 summarizes this approach for analysing differences in trace recordings based

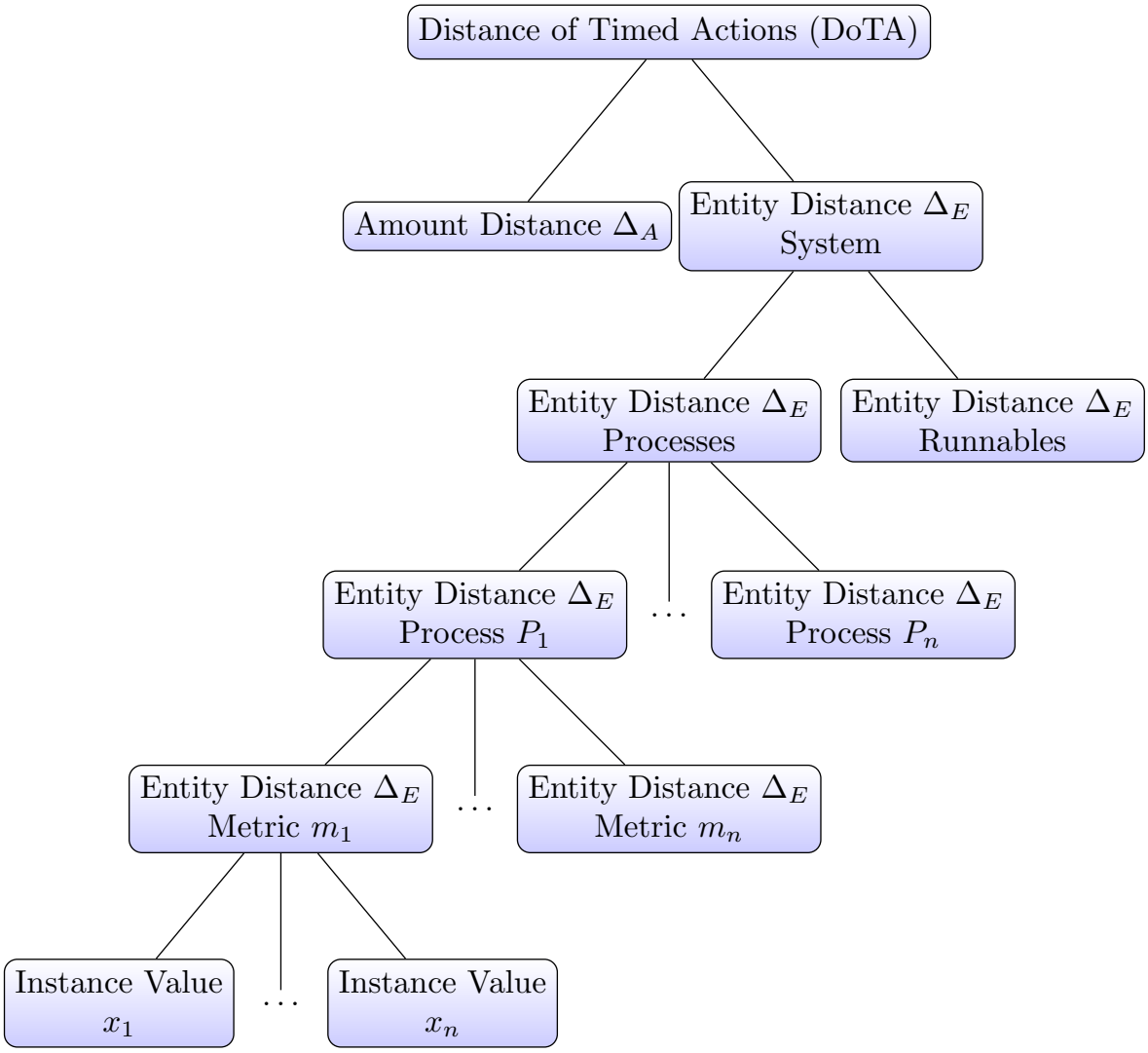


Figure 7.6.: DoTA Approach. Visualisation of the multi-level approach for analysing differences in trace recordings based on the measure Distance of Timed Actions.

on our measure DoTA.

7.4. Validation

In Chapter 7.1, we present the challenges of comparing real-time behaviour based on trace recordings. To do so, we introduce a set of models that represent common architectural patterns in the real-time software domain and feasible variations for each pattern that provoke realistic changes to a system's timing behaviour. Quantifying the impact of such a variation states the challenge that a suitable measure has to meet in a reasonable way.

To show the expressiveness of our measure and to highlight the impact of the changes made by each variation, we determine how traces of each variation differs from traces of the previous one using *Distance of Timed Actions (DoTA)*. In addition, we also consider Huselius' *Sum of Divergence*, an existing measure from the latest related work as presented in Chapter 7.2. For conducting this exemplary evaluation, we generate a simulation trace that covers 100 s of system execution for each model using the TA Simulator [70]. The results for both measures, which are listed in detail in Tab. 7.2, are visualised in Fig. 7.7.

Table 7.2.: Differences in Trace Recordings from Variations of different Architectural Patterns.

Variations	1 vs. 2	2 vs. 3	3 vs. 4	4 vs. 5	5 vs. 6	6 vs. 7	7 vs. 8
Purely Periodic without Communication							
DoTA	31.3	31.1	25.0	15.3	0.7	8.5	10.9
SoD	32.1	65.5	41.5	22.6	9.4	18.0	24.0
Client Server without Reply							
DoTA	45.9	50.8	28.4	6.0	1.2	8.9	
SoD	9.6	14.5	58.3	0.1	0.2	0.1	
State Machine							
DoTA	42.2	42.2	13.3	5.8	0.1	10.4	
SoD	4.4	6.0	7.1	0.3	5.0	0.7	
Feedback Loop							

Variations	1 vs. 2	2 vs. 3	3 vs. 4	4 vs. 5	5 vs. 6	6 vs. 7	7 vs. 8
DoTA	49.7	16.1	19.9	11.3	3.2	1.6	
SoD	15.6	7.7	21.7	15.0	11.0	8.4	
State Machine Feedback Loop							
DoTA	66.0	20.1	17.0	12.6	3.5	0.8	
SoD	19.9	11.5	15.7	20.5	14.8	12.2	

The results of DoTA for the first three variations of the system pattern *'Purely Periodic without Communication'* proceed nearly unchanged. This is because each variation modifies the system in the same way, namely by adding one more task. Instead, the results of Huselius' Sum of Divergence alternate massively. The low difference between Variation 4 and Variation 5 is also consistent because the queue overflows eliminated by the increase of the maximum number of queued activation requests in Variation 5 happen only in very few situations.

The variations for all other system patterns are equal. This is noticeable in the depicted results of DoTA but not in those of Huselius' Sum of Divergence. Because each pattern contains data dependencies, the exclusive areas added by Variation 2 affect not only all tasks directly but also indirectly via arising blocking situations, which, then, results in the high difference at the beginning. The results of DoTA stay high for the next variation of the system patterns *'Client-Server without Reply'* and *State Machine*. Both patterns consist of only two tasks and one of these tasks is modified by adding an interprocess activation. However, the other patterns consist of multiple tasks which is why the variation has a lower impact on the overall system. All other variations result in differences between 0 % and around 20 % and are, thus, also corresponding to the results for the system pattern *'Purely Periodic without Communication'*.

7.5. Other Use Cases

Although Distance of Timed Actions (DoTA) is motivated by assessing how closely a synthesised model reflects the timing behaviour of the underlying system, it has multiple other applications. Two use cases to which we have successfully applied our measure are presented in the following

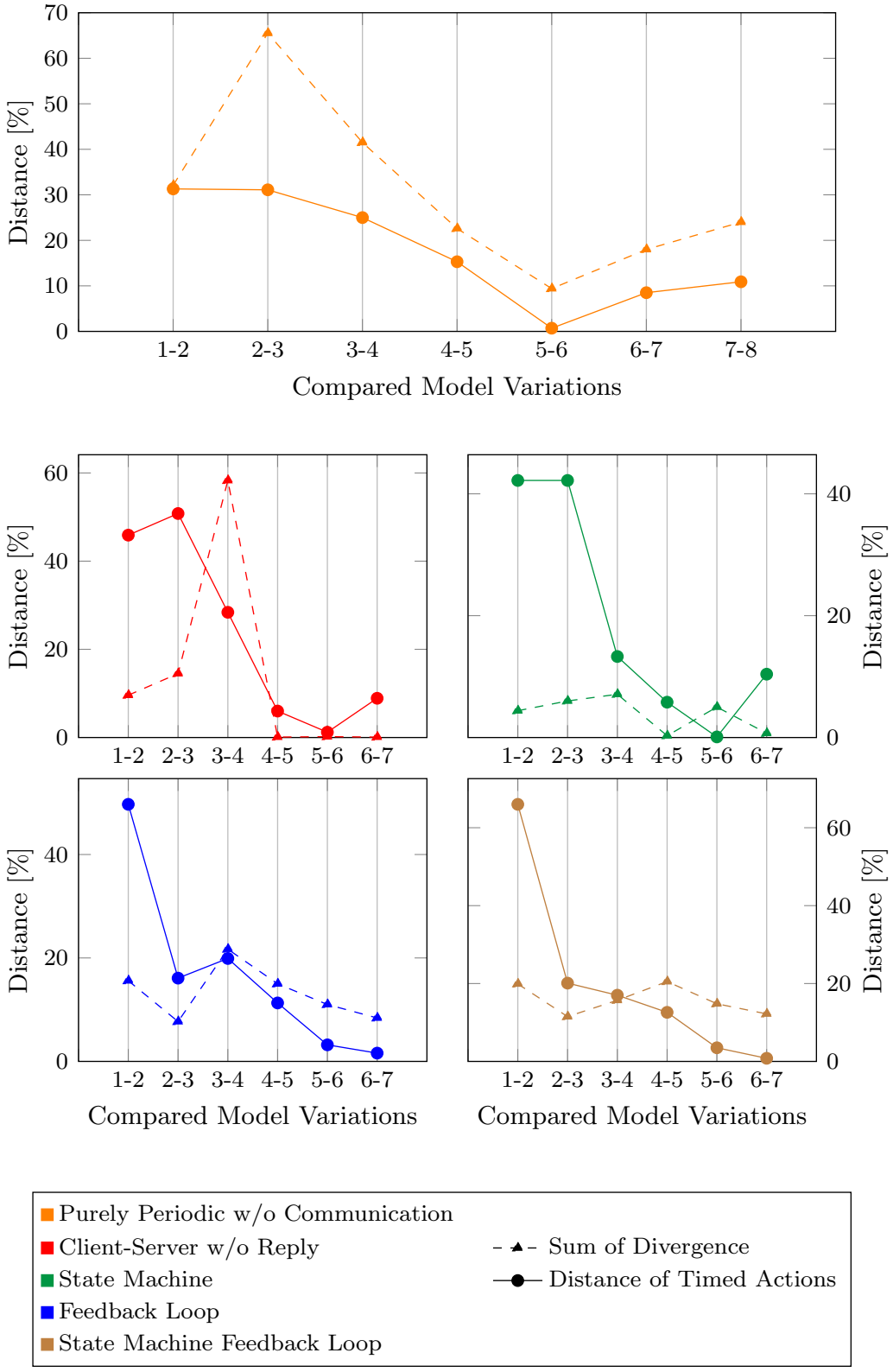


Figure 7.7.: Validation Results for DoTA. Each marker denotes the differences determined by our measure Distance of Timed Actions (DoTA), resp., Huselius' Sum of Divergence between trace recordings of two succeeding model variations. The different colours mark the individual system architecture patterns examined in our validation. Adapted from [13].

7.5.1. Product Family

One use case to which we have successfully applied our measure is determining the effects of system changes on the system's timing behaviour. To explain this, assume

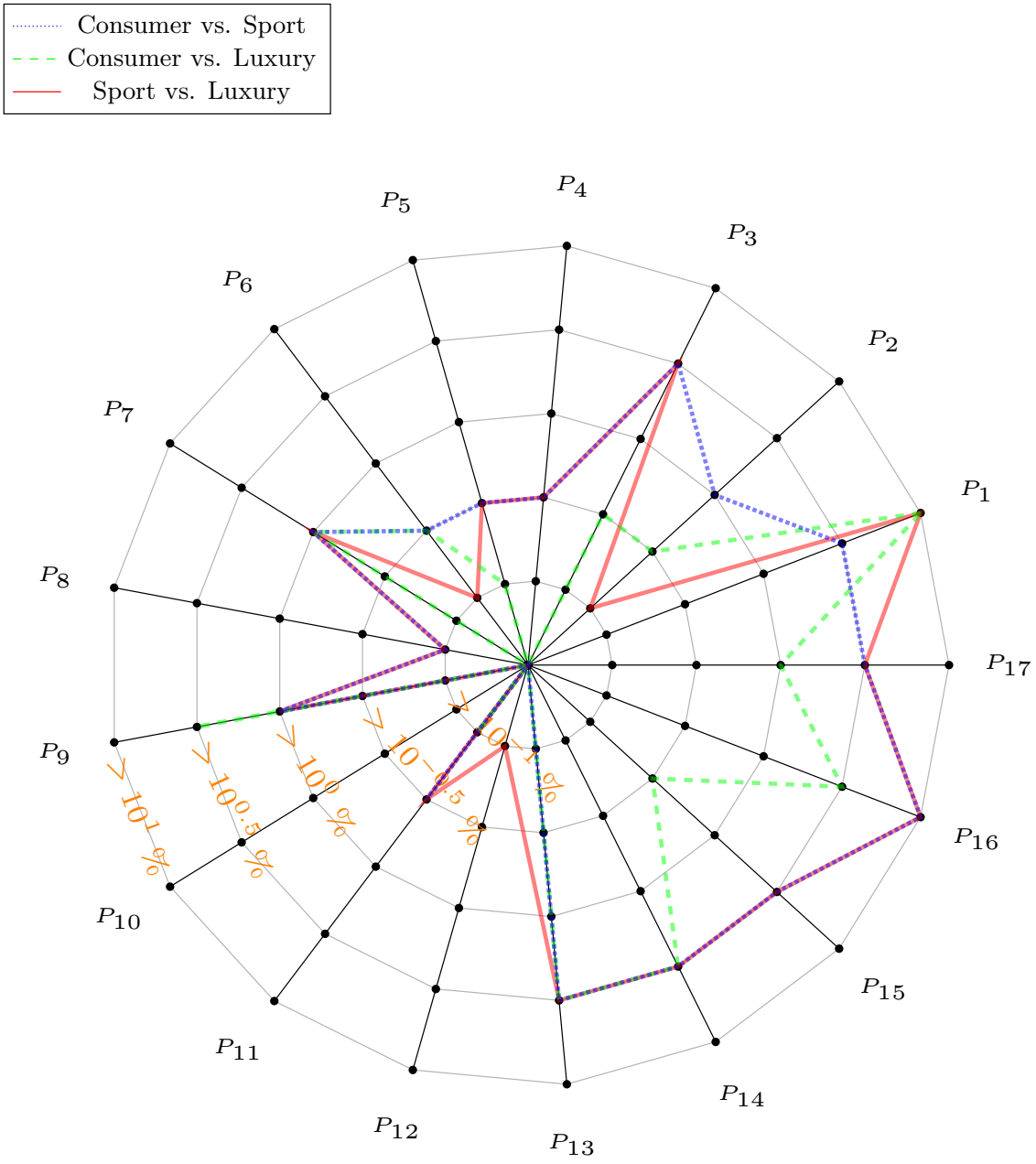


Figure 7.8.: Results for Use Case 'Product Family'. Radar chart showing the differences of each process (tasks and ISRs) in case of comparing the products 'Consumer', 'Sport', and 'Luxury' with each other. Each process of the common software product platform is represented by a spoke on which the change is plotted based on our measure, and each colour indicates the products of the product family that are compared. Reprinted from [13].

not a single product but a product family for an industrial steering system, so that it is possible to create software for the three distinct products 'Consumer', 'Sport' and 'Luxury', which all have the same architecture but different functionalities. The steering system consists of 17 tasks and ISRs, which together call 130 different runnables. The employed hardware platform is the same for all three products and consists of a dual core processor with a frequency of 120 MHz for each processing unit.

To analyse the impact of the varying functionalities on the timing behaviour of the system's processes (tasks and ISRs), we have compared, in pairs, trace recordings of the products using our measure. Each trace recording covers 30 s and contains all task and runnable calls performed during that time. This resulted in roughly $27 \cdot 10^6$ events and a file of 1.7 GB, i.e., the system produces roughly one million events per second. The results of our comparison, which are listed in detail in Tab. 7.3, are depicted in Fig. 7.8 and show that not all processes are affected to the same extent by a switch ta a different product of the product family, e.g., process P_{10} has exactly the same timing behaviour in all three products, whereas the timing behaviour of processes P_1, P_3, and $P_{13} - P_{17}$ differ widely, with differences partially over 10 %.

Table 7.3.: Differences between the processes for comparing trace recordings from different products using DoTA.

	Sport vs. Luxury [%]	Consumer vs. Luxury [%]	Consumer vs. Luxury [%]
P1	11.56429082	10.45917769	6.351332577
P2	0.300603748	0.930040196	1.159205268
P3	4.69030997	0.855764358	4.122411453
P4	0.344240474	0.008929258	0.344180975
P5	0.544844983	0.137329436	0.487636673
P6	0.274527131	0.702967963	0.503836284
P7	1.657822857	1.625155677	1.403627162
P8	0.252377514	0.094432932	0.223618758
P9	2.671538317	3.675472482	1.337137081
P10	0.077832454	0.099147329	0.0582255
P11	0.50450458	0.263378962	0.394792842
P12	0.103067698	0.099147329	0.051912505

Process	Sport vs. Luxury [%]	Consumer vs. Luxury [%]	Consumer vs. Luxury [%]
P13	5.65491727	3.17575714	5.033000926
P14	7.399536903	9.11219579	9.555831633
P15	4.045246556	0.885569202	3.970209795
P16	13.65540227	5.399524851	14.33665408
P17	9.059978589	1.782551008	8.729271495

7.5.2. Trace Check

Another use case to which we have successfully applied our measure is assisting with hardware tracing. The process of configuring the on-chip debug units is known to be error-prone. For that reason, we use Distance of Timed Actions (DoTA) to highlighting inconsistencies in the recorded system behaviour.

The main problem with an incorrectly configured on-chip debug unit is that ob-

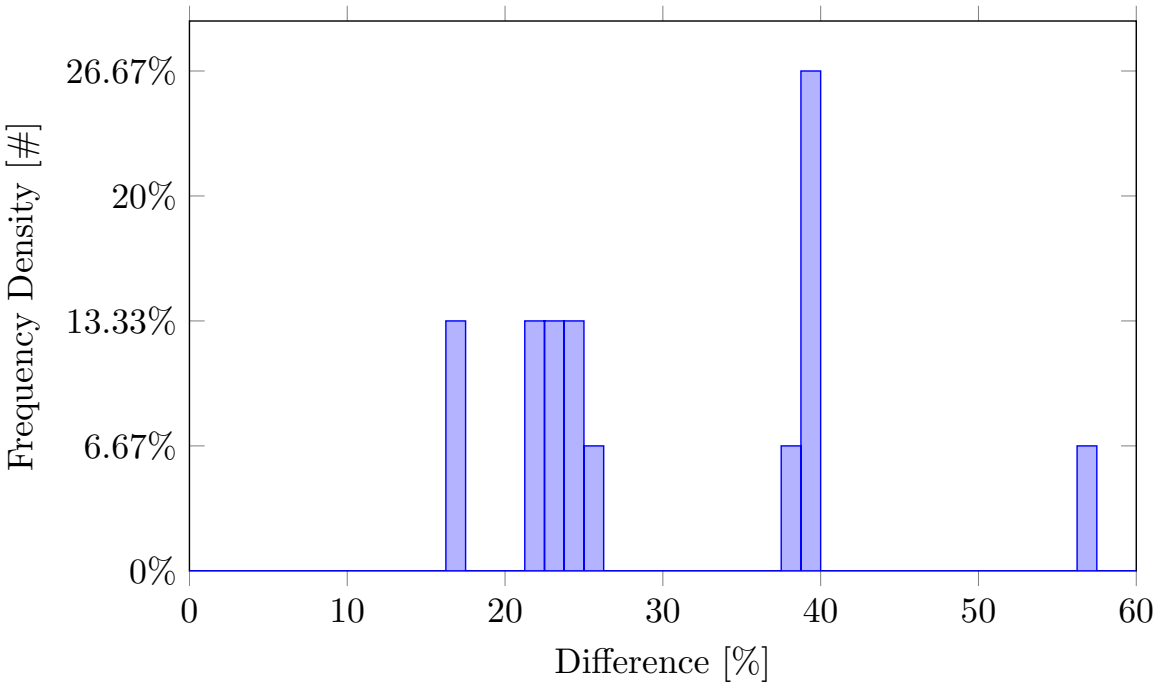


Figure 7.9.: Results for Use Case 'Trace Check'. Histogram summarising the correspondence of a 'clean' trace and a trace that contains errors because of an incorrectly configured on-chip debug unit. The blue bars indicate the resulting differences of the processes (tasks and ISRs).

served events in the trace buffer are overwritten before the information is transferred to the host. As a consequence, events are missing sporadically. Because the system evolves through well defined states during execution, the missing of events leads to invalid state transitions. To compensate this and to get a valid trace, events are added at the required positions to satisfy the underlying state machines. Thus, the trace looks good at first sight but reflects in some situations an unlikely timing behaviour of the underlying system because of the incorrect timestamps of the added events. As a consequence, we apply DoTA after changing the debug configuration to check if the recently recorded trace reflects the same timing behaviour. The histogram depicted in Fig. 7.9 shows the results of such a comparison with a trace that contains errors because of an incorrectly configured on-chip debug unit.

This use case origins from an experience with tracing an industrial braking system. The system consists of 15 tasks and ISRs, which together call 89 different runnables. At first, we configured the on-chip debug unit to get a trace recording at process level. Then, we intended to change the configuration in such a way that the trace also contains runnable events. In the course of this, we got something wrong, which resulted in the aforementioned buffer-overflows and the incorrect trace recordings. Because the trace looked good at first sight, the errors were not detected until a few hours later when the system was analysed in detail and some system characteristics turned out to be inconclusive. As a consequence, we applied DoTA to compare the trace on process level with the recently recorded trace and found out that all processes clearly differ with differences up to 57 % as listed in Tab. 7.4 and depicted in Fig. 7.9.

Table 7.4.: Differences between the processes for comparing a 'clean' trace and a trace that contains errors using DoTA.

Process	DoTA [%]	Process	DoTA [%]	Process	DoTA [%]
P1	22.26	P6	23.52	P11	25.74
P2	16.30	P7	37.86	P12	39.21
P3	38.99	P8	57.24	P13	39.33
P4	16.86	P9	22.42	P14	24.27
P5	22.83	P10	39.09	P15	24.83

8

Evaluation

One motivation for us to propose a novel measure is to assess the quality of our reverse engineering tool, CoreTAna. This is done in the following in three different ways. At first, the reverse engineering tool under evaluation has to manage realistic challenges of the real-time development process in a synthetic benchmark. This determines how well the intended field of applications is handled. Next, we apply CoreTAna to randomly generated systems to explore its performance in the broad design space of real-time systems. Finally, a tighter connection to real-life problems is established by using CoreTAna in actual industrial projects.

8.1. Synthetic Benchmark

The goal of this benchmark is to provoke realistic challenges of the real-time development process, e.g., blocking situations, which then have to be managed by a reverse engineering tool under evaluation. For that reason, we use the models and variations defined as the challenges of comparing real-time behaviour from trace recordings in Chapter 7.1 as the basis of this synthetic benchmark.

To be able to conduct an exemplary evaluation on measuring the performance and the quality of CoreTAna's reverse engineering, we generate a simulation trace that covers a specified number of time units for each system using the TA Simulator [70]. The numbers are chosen in such a way that in the first case only a few process activations are recorded and in the other case an extensive amount of information is sampled. Further, trace recordings are generated at both process level, which is limited to process events, and system level, which provide a detailed insight into a system's behaviour. The traces are then analysed by CoreTAna, which is included in the TA Tool Suite Release 16.3 [70]. This analysis is performed on a workstation containing an Intel Core i7-4930K hexa-core CPU, where each core is clocked at a frequency of 3.4 GHz, with 32 GB of RAM and running the 64-bit version of Windows Server

2012.

To manifest a high quality of the performed reverse engineering, our measure has to yield results close to zero percent when comparing the trace recordings of the actual system with those generated when simulating the synthesised model. Mismatches in the recorded behaviour are revealed by our measure due to the fact that the system's scheduling propagates each disparity on and on, and due to the quadratic characteristic of the used Euclidean distance. We also determine Huselius' Sum of Divergence, not only to compare it with our measure but also to substantiate its shortcomings mentioned in Chapter 7.2.

8.1.1. Purely Periodic without Communication

For each of these systems a trace recording is applied to CoreTAna and the difference to those generated when simulating the synthesised model is determined using our measure Distance of Timed Actions (DoTA) and Huselius' Sum of Divergence. The results, which are listed in detail in Tab. 8.1, are visualised in Fig. 8.1 and turn out to be two fold in case of our measure. If all system characteristics are supported by CoreTAna's underlying reverse engineering approach, the synthesised model reflects the actual system behaviour very well. This is demonstrated by the high similarity between the trace recordings of Variations 1 to 6 in the figure. However, if at least one system characteristic, such as the scheduling algorithm, is not considered in CoreTAna, then the difference rises rapidly. This is due to the fact that even if only one task is affected by a variation, this change can have an impact on all tasks because of the scheduling. Moreover, the Euclidean distance used in our measure has a quadratic characteristic.

In contrast, the gap between Variation 7 and 8 is far less distinct when determining the trace differences using the Sum of Divergence. Furthermore, Huselius' measure manifests in general quite pessimistic results for this system architecture pattern with most results being around 20 percent. This corresponds to the outcome of the determined impact of change between the individual variations (see Fig. 7.7), where the Sum of Divergence also shows a higher disparity between the traces than using DoTA.

Noticeable is the fact that the results of the reverse engineering for the Variations 1 to 6 are lower by a factor 10 if the longer trace recording with the added information is used. This means, that the results from the short trace are too pess-

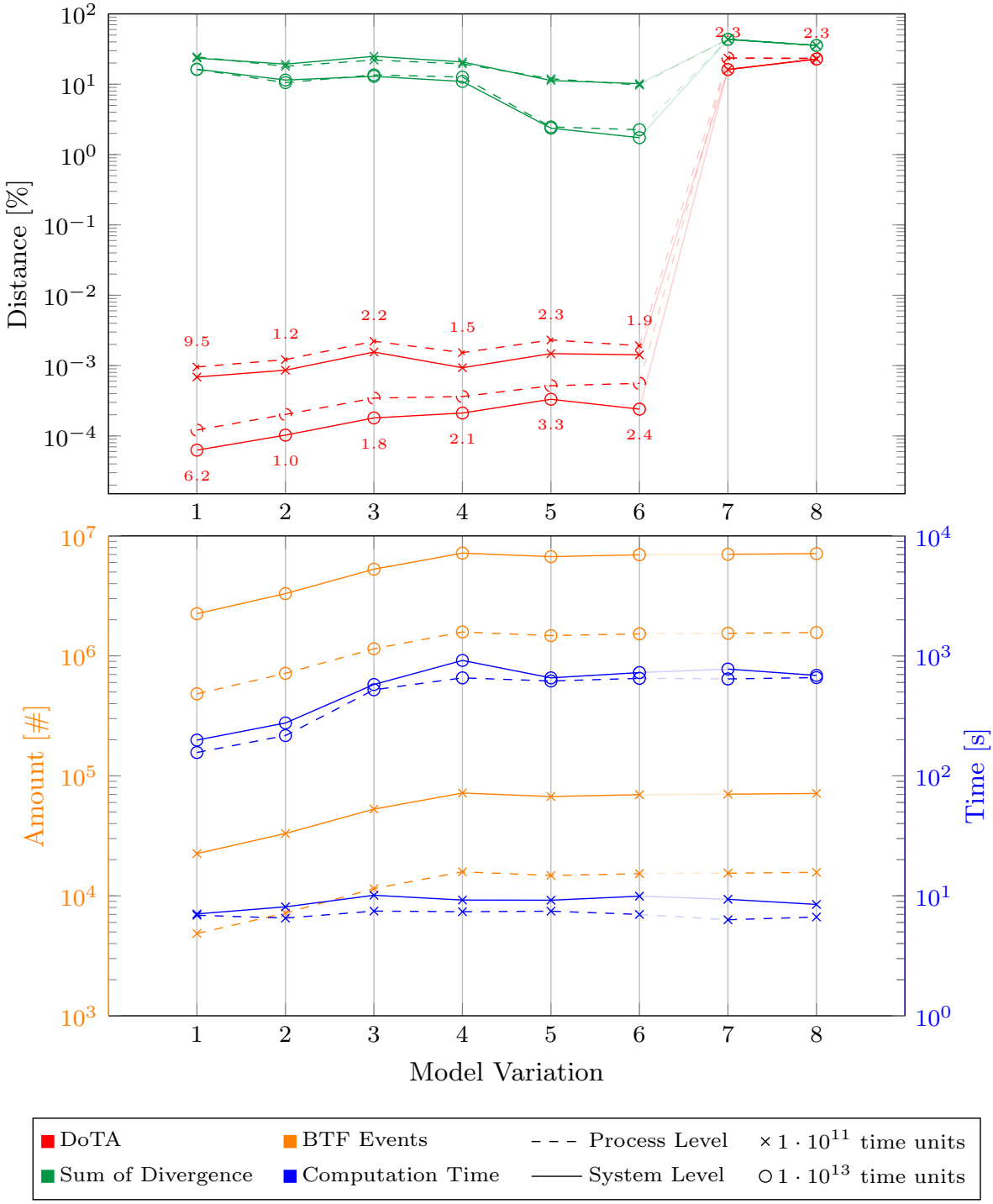


Figure 8.1.: Results of CoreTAna for the variations of system architecture pattern 'Purely Periodic without Communication'. Each red mark, resp., green mark denotes the result of our measure Distance of Timed Actions, resp., Huselius' Sum of Divergence for comparing a trace of the pattern, resp., its variation with one generated when simulating CoreTAna's reversely engineered model. The reduced opacity in the lines isolates variations that feature characteristics that are not supported by CoreTAna (Schedule Points and EDF Scheduling). The orange marks, resp., blue marks indicate the number of events in the traces of the pattern, resp., the time it took CoreTAna to synthesise the model from the trace.

imistic and that they are actually closer to zero.

The runtime that CoreTAna takes to reconstruct a model is depicted on the lower chart of Fig. 8.1 and appears to increase linearly with the number of events within the trace recording. This due to the design of the reverse engineering approach, as presented in Chapter 6.2, where the algorithms are designed in such a way that the complete trace recording and, thus, all trace events are processed in chronological order exactly once. Noticeable is that the runtime contains a constant part, e.g., accessing the trace database, since the runtime is steady for shorter trace recordings although the number of events within the trace doubles from Variation 1 to 8.

Fig. 8.1 also shows that there is not much difference between the results of the reverse engineering from traces at process level and those at system level. This is due to the fact that this system pattern does not contain any data dependencies and, thus, no information besides the function calls is added at system level. Just the time it takes to perform the reverse engineering is differently because the added function calls at system level have to be processed.

Table 8.1.: Detailed Results of CoreTAna for 'Purely Periodic without Communication'.

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Distance of Timed Actions [%]	1	0.0009537	0.0001211	0.0006882	0.00006277
	2	0.0012154	0.0002020	0.0008599	0.00010262
	3	0.0022185	0.0003446	0.0015576	0.00018022
	4	0.0015281	0.0003633	0.0009322	0.00021145
	5	0.0023127	0.0005166	0.0014787	0.00033187
	6	0.0019142	0.0005592	0.0014220	0.00024032
	7	23.513359	23.480054	16.174810	16.1866607
	8	23.282120	23.231606	22.774799	22.8086669

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Sum of Divergence [%]	1	24.4914285	16.242624	23.5200695	16.359409
	2	17.9727093	10.541485	19.2979265	11.465052
	3	22.0566686	13.565652	24.9350911	12.943896
	4	19.4783329	12.639322	20.7283790	10.955535
	5	11.9588580	2.4720624	11.3080650	2.3752584
	6	9.73254768	2.2561763	10.1844009	1.7398719
	7	43.5854961	43.394641	43.5800288	43.393426
	8	35.5840176	35.599870	35.5733129	35.605577
Amount of BTF Events [#]	1	4852	482252	22489	2246689
	2	7185	715310	33083	3303933
	3	11489	1145864	52811	5276161
	4	15848	1581973	71927	7187277
	5	14782	1475307	67231	6717281
	6	15290	1526415	69695	6965045
	7	15454	1542529	70351	7029501
	8	15688	1565863	71287	7122847
Computation Time [s]	1	6.8899295	156.53972	7.0686523	198.57825
	2	6.5212860	216.66097	8.0940568	275.82592
	3	7.4617857	521.61375	10.105386	577.56854
	4	7.3750913	655.67053	9.2230262	916.49947
	5	7.4464685	617.86835	9.1913158	655.81481
	6	6.9981298	649.62631	9.9317691	725.65135
	7	6.3163664	642.55496	9.3535587	776.39202
	8	6.6608028	659.24892	8.4779536	687.60256

8.1.2. Client-Server without Reply

Fig. 8.2 depicts the results of our measure Distance of Timed Actions (DoTA) and Huselius' Sum of Divergence for the variations of the system pattern 'Client-Server without Reply', which are listed in detail in Tab. 8.2. The first thing that leaps out compared to the outcome of the previous system pattern is the gap between DoTA's results from the traces at process level and those at system level. In Fig. 8.1, both corresponding red lines run next to each other and for this system pattern they differ from each other roughly by a factor of 100. This difference is due to the fact that the data dependencies added by this system architecture pattern happen within the context of a task. Since the generated trace recordings at process level are limited to task events, the internal behaviour of tasks is totally unapparent to CoreTAna. This black box like behaviour of tasks makes it impossible to deduce a more precise model without the expense of adding uncertainty. The results of Huselius' Sum of Divergence show the same behaviour, although they are not as clear because of the measure's volatility.

The reduced amount of information contained within the traces at process level entails also that only a few events represent all the available information and no large variability exits. As a consequence, the results lie side by side no matter how long the recorded time frame of the system is.

On closer examination of the results, another consequence of the increased complexity of this system pattern when compared to the previous one stands out. This manifests itself in the fact that the values resulting from our distance measure DoTA differ by a factor of 100 for traces at system level and approximately by a factor of 10,000 for traces at process level (see red marks around 10^{-1} %, resp., 10^{1} % in Fig. 8.2 vs. those around 10^{-4} %, resp., 10^{-3} % in Fig. 8.1).

Although the results produced by Huselius' Sum of Divergence give, again, a more pessimistic evaluation of CoreTAna's capabilities than DoTA does, they are roughly in line with each other except for Variation 2 and 3. There, the distance between the trace recording of the actual system and those generated when simulating the synthesised model increases from the first variation in case of our measure, but the Sum of Divergence decreases. The main reason for this is that Huselius' measure is normalised by the maximum difference between the trace recordings [9, p.143 ff]. That way, outlier results that occur infrequently distort the measure's significance about the general behaviour. The impact of small changes between the individual variations as depicted in Fig. 7.7 show a similarly volatile behaviour.

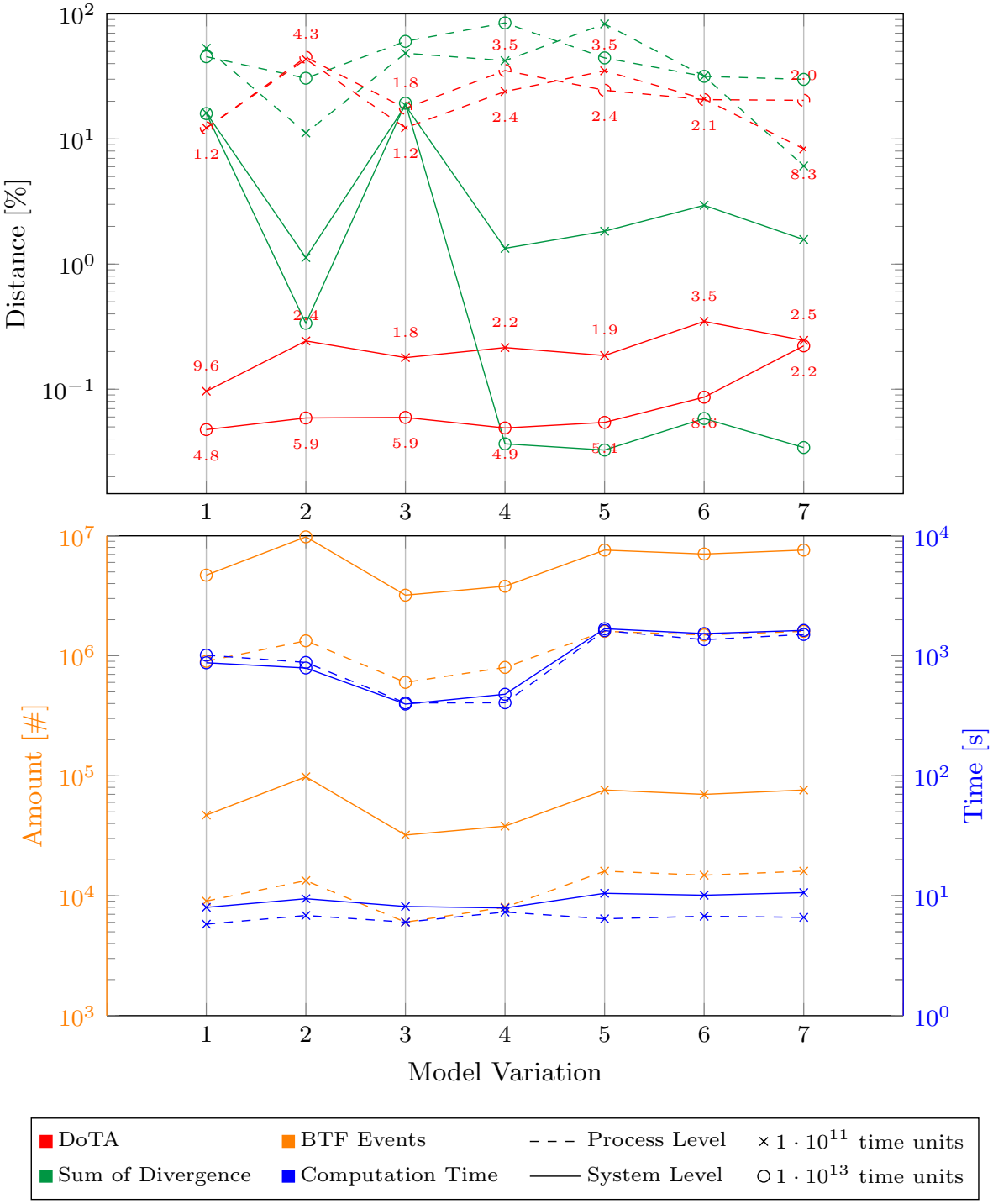


Figure 8.2.: Results of CoreTAna for the variations of system architecture pattern 'Client-Server without Reply'. Each red mark, resp., green mark denotes the result of our measure Distance of Timed Actions, resp., Huselius' Sum of Divergence for comparing a trace of the pattern, resp., its variation with one generated when simulating CoreTAna's reversely engineered model. The orange marks, resp., blue marks indicate the number of events in the traces of the pattern, resp., the time it took CoreTAna to synthesise the model from the trace.

CoreTAna's runtime to reconstruct a model is in line with those from the previous system pattern. Small fluctuations in the amount of events have hardly an impact on the computation time, while the time it takes CoreTAna to process larger trace recordings increases roughly linearly with the number of events. In general, the amount of events in the trace and the corresponding amount of time it takes CoreTAna to process this trace is around five times higher than in the previous system pattern because of the added signal accesses.

Table 8.2.: Detailed Results of CoreTAna for 'Client-Server without Reply'.

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Distance of Timed Actions [%]	1	12.256728	12.236614	0.0963846	0.0475689
	2	43.361206	44.942581	0.2431276	0.0589234
	3	12.275972	17.572060	0.1788738	0.0595190
	4	23.898499	35.169433	0.2151218	0.0490214
	5	35.011617	24.478762	0.1859166	0.0543016
	6	20.851929	20.610418	0.3481599	0.0863580
	7	8.3254603	20.343858	0.2457119	0.2220823
Sum of Divergence [%]	1	53.130927	45.400173	16.138064	15.940925
	2	11.159526	30.551754	1.1255763	0.3363400
	3	48.423795	60.213025	18.491527	19.290745
	4	42.222291	84.589280	1.3349835	0.0366133
	5	83.016100	44.417854	1.8302575	0.0326665
	6	31.746164	31.615589	2.9489597	0.0583671
	7	6.0848614	29.916530	1.5705853	0.0342065

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Amount of BTF Events [#]	1	9020	900020	47017	4700017
	2	13355	1333353	98031	9800015
	3	6020	600020	32017	3200017
	4	8020	800020	38017	3800017
	5	16020	1600020	76017	7600017
	6	14830	1492899	69897	7049109
	7	16018	1600020	76008	7600017
Computation Time [s]	1	5.7949492	1012.0734	7.9979248	874.56971
	2	6.8443415	880.50990	9.4392821	791.23004
	3	6.0229296	404.26895	8.1328763	395.94720
	4	7.3102416	406.10139	7.9014146	477.79289
	5	6.4200232	1620.3791	10.467379	1680.8327
	6	6.7426784	1362.8241	10.092693	1531.5093
	7	6.6013917	1505.8412	10.604121	1628.2769

8.1.3. State Machine

The results for the system pattern 'State Machine', which are listed in detail in Tab. 8.3 and visualised in Fig. 8.3, show a noticeable similarity to those for the previous pattern. Our measure Distance of Timed Actions (DoTA) yields also results at around 20 % for reverse engineering a model from the trace recordings at process level and differences less than 0.5 % at system level. Just the difference between the peaks and valleys of the results for DoTA become greater, especially for the shorter trace recordings. This is due to the characteristics of the probabilistic model used in this system architecture pattern. Recording the system's runtime behaviour for a longer period and employing those to CoreTAna yields steadier results, because shorter trace recordings have a smaller chance to cover all possible behaviour of the probabilistic model. Because the aforementioned probability is hardly observable in trace recordings at process level, both dashed red lines nearly overlap which means that recording

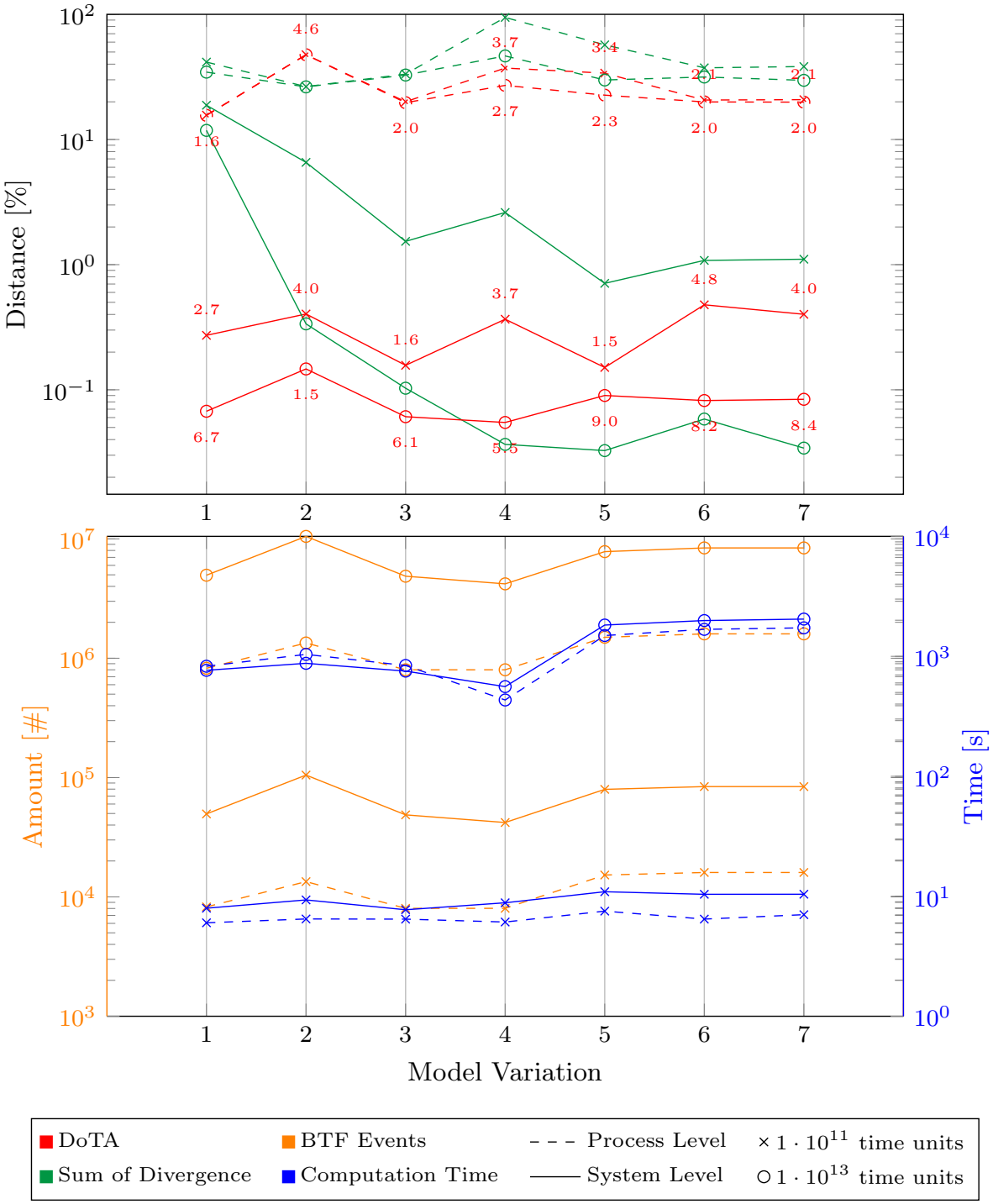


Figure 8.3.: Results of CoreTAna for the variations of system architecture pattern 'State Machine'. Coloured marks and line styles are used as in Fig. 8.2.

events for a longer period of time than in the short trace recording does not add information.

The results of Huselius' Sum of Divergence are similar to those of DoTA. The course of the red and green dashed lines, i.e., the values for the trace recordings at process level, lie on the same level. Also, a similarity between the lines of the short trace at system level is noticeable. Just the continuous drop from the first to the third variation differs the results from those of DoTA. This drop is even more developed at the results from the trace recordings at system level that last 10^{13} time units because of the aforementioned volatility of the measure. Like it is shown in the results of the previous system pattern in Fig. 8.3, only the results of Huselius' Sum of Divergence of the first three variations are distant. The remaining values are in line with those of DoTA.

The amount of events resulting from simulating the model and the time it takes CoreTAna to process these events are nearly unchanged compared to the previous system model. The results show that CoreTAna can process roughly 10,000 events per second in case of a trace recording at system level as input and around 1000 events when analysing a trace recording at process level. The smaller throughput of events in the latter case is because CoreTAna tries to deduce information that is missing in the trace recordings at process level which takes additional time. A noticeable difference to the results of the previous system patterns is that the ratio of amount of events that are processed per time unit is very low for the long trace at process level which manifests in such a way that both the blue line and orange line are nearly congruent.

Table 8.3.: Detailed Results of CoreTAna for 'State Machine'.

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Distance of Timed Actions [%]	1	15.652798	15.580206	0.2723777	0.0673689
	2	47.586166	47.533505	0.4033630	0.1467363
	3	20.032283	19.697544	0.1569058	0.0609489
	4	37.221495	27.055662	0.3663406	0.0548155
	5	33.893068	22.612843	0.1508038	0.0900652

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
	6	20.664853	19.964286	0.4777384	0.0820607
	7	20.817826	19.882330	0.4011350	0.0839657
Sum of Divergence [%]	1	41.551175	34.541668	18.781859	11.821757
	2	26.440948	26.323879	6.5627732	0.3363400
	3	33.312319	32.728243	1.5343503	0.1029537
	4	94.466047	46.515561	2.6116502	0.0366133
	5	57.039978	29.888022	0.7087619	0.0326665
	6	37.397598	31.573412	1.0810095	0.0583671
	7	38.277515	29.698478	1.1049296	0.0342065
Amount of BTF Events [#]	1	8245	823835	49587	4961947
	2	13431	1341983	104928	10492584
	3	8021	800021	48691	4866691
	4	8020	800020	42018	4200018
	5	15255	1498527	79647	7820058
	6	16020	1599908	84018	8399378
	7	16020	1599803	84018	8398778
Computation Time [s]	1	6.0107281	826.54534	7.9558477	765.18599
	2	6.4716200	1038.2568	9.3250706	874.84334
	3	6.4486606	838.82497	7.7223145	753.26378
	4	6.1018826	432.66732	8.8268628	559.27888
	5	7.5224902	1490.7532	10.911711	1822.8462
	6	6.4538438	1678.3919	10.386822	1987.4026
	7	7.0271813	1728.0191	10.398466	2043.5359

8.1.4. Feedback Loop

Fig. 8.4 visualises the resulting quality of the performed reverse engineering for the system pattern 'Feedback Loop', which is listed in detail in Tab. 8.4. Because this system pattern is more complex than the previous one, the results of our measure Distance of Timed Actions (DoTA) using trace recordings at system level are slightly worse. However, those for trace recordings at process level are noticeably better with values mainly between 14 % and 17 %. The reason for this is that periodically activated tasks without communication are added in this system pattern in order to increase the complexity, which are reverse engineered by CoreTAna quite good from trace recordings at process level as it is shown in Fig. 8.1. Noticeable is also that the length of the trace recording at process level does not have a big impact on the resulting quality of the reverse engineering which manifests in such a way that the red dashed lines are nearly congruent. This means that CoreTAna cannot recover the missing information from system level more precisely even if a trace recording is used as input that covers the system's execution for a longer period of time.

This time also the results of the trace recordings at system level show significant spread, which can once again be explained by the underlying probabilistic model. Also the distance between the results of DoTA for the trace recordings at system level covering a short period of time and those that last 10^{13} time units lie wide apart. Because the complexity of the system increased further as in the previous system architecture, the chance to cover all possible behaviour decreased to such an extent that even large trace recordings do not cover everything.

The results of Huselius' Sum of Divergence show a big difference compared to those from the previous system pattern. In Fig. 8.3, the results for the trace recordings at process level and those at system level lie clearly separated from each other with a big gap of 50 % to 100 % in between. Although the results are not as volatile as before, they nearly run next to each other in Fig. 8.4. Thus, Huselius' Sum of Divergence yields results that are not only very pessimistic but also that conclude that adding pieces of information about the system level does not have an impact on the quality of the reverse engineering of CoreTAna.

The amount of events resulting from simulating the model and the time it takes CoreTAna to process these events show no significant difference in comparison to the previous system model.

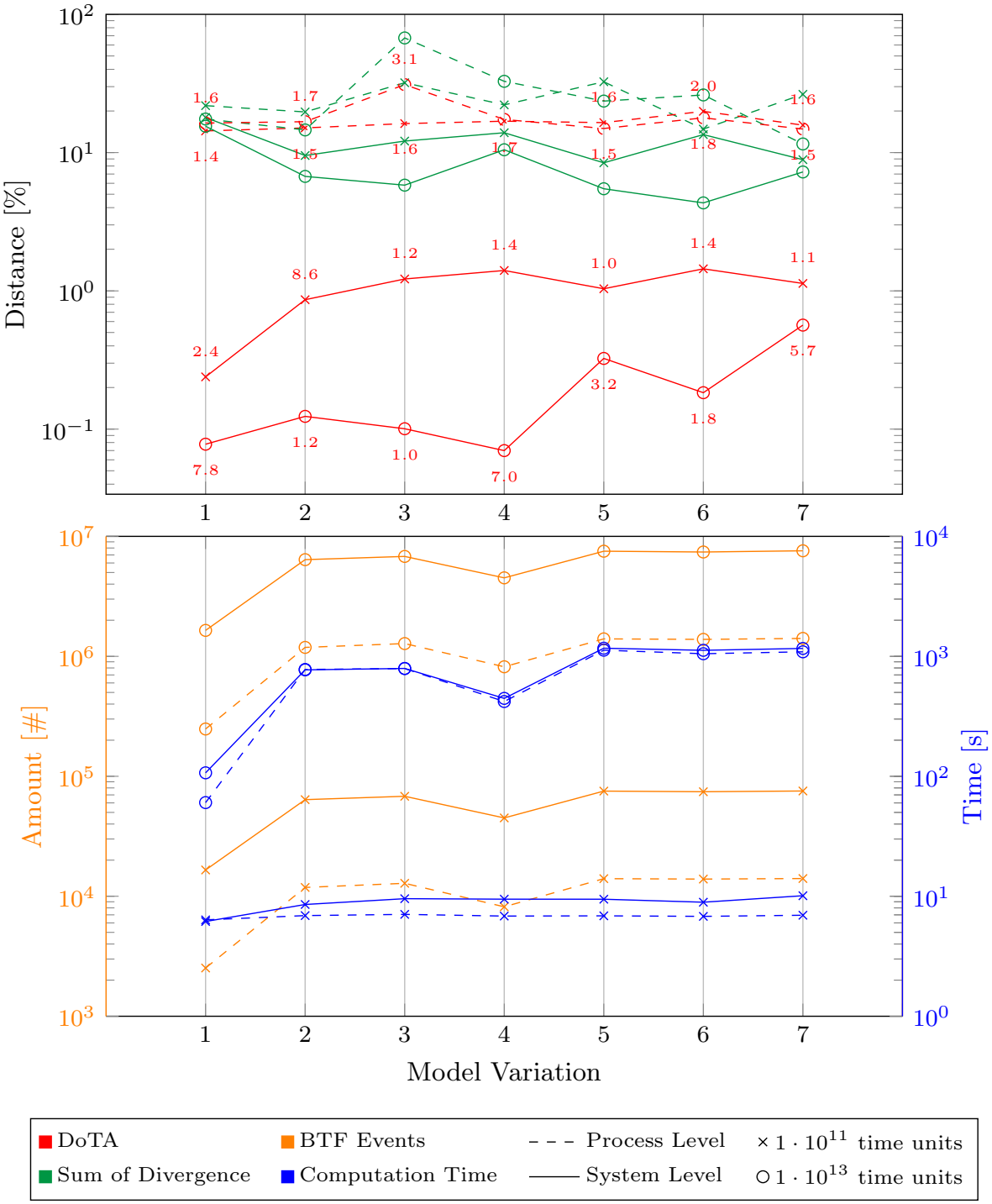


Figure 8.4.: Results of CoreTAna for the variations of system architecture pattern 'Feedback Loop'. Coloured marks and line styles are used as in Figs. 8.2 & 8.3.

Table 8.4.: Detailed Results of CoreTAna for 'Feedback Loop'.

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Distance of Timed Actions [%]	1	14.349606	16.358382	0.2388283	0.0778398
	2	15.091768	16.735423	0.8633648	0.1237109
	3	16.216500	31.022230	1.2190268	0.1006605
	4	16.874797	17.374592	1.4043835	0.0699575
	5	16.480638	14.940903	1.0361191	0.3246640
	6	19.889402	17.850104	1.4437576	0.1835005
	7	15.796890	14.808603	1.1318237	0.5654531
Sum of Divergence [%]	1	21.861669	17.499155	17.952231	15.516433
	2	19.689258	14.589903	9.5256064	6.7289407
	3	32.015413	67.512619	12.118766	5.8157202
	4	22.148177	32.783324	13.920895	10.487218
	5	32.547768	23.588531	8.4394832	5.4783079
	6	14.801654	26.126691	13.496559	4.3285223
	7	26.397922	11.546924	8.8943718	7.2466639
Amount of BTF Events [#]	1	2524	247935	16569	1644865
	2	11847	1186978	63889	6395294
	3	12826	1277567	68147	6803720
	4	8171	818247	44992	4509812
	5	14023	1397968	75287	7521149
	6	13908	1384217	74363	7406301
	7	14059	1410331	75431	7581103

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Computation Time [s]	1	6.3590104	60.396155	6.1630692	106.90990
	2	6.9041974	777.51649	8.5426437	771.42144
	3	7.0657849	790.93414	9.5481376	792.74775
	4	6.8400576	420.28964	9.4596645	446.21983
	5	6.8709635	1125.7334	9.4511110	1168.8052
	6	6.8017730	1051.3514	8.9236850	1122.4087
	7	6.9486556	1090.7730	10.106129	1163.6606

8.1.5. State Machine Feedback Loop

The resulting quality of the performed reverse engineering for the system pattern "State Machine Feedback Loop", which is listed in detail in Tab. 8.5, is visualised in Fig. 8.5. Because of the periodically activated tasks without communication that are added in this system pattern, the results, which are mainly between 11 % and 22 %, are quite low, again, when trace recordings at process level are used as input.

Noticeable is, furthermore, the fact that the results for both trace lengths lie close upon each other. Just the lines for trace recordings at system level have still a big gap between them, which, however, is smaller than in the previous system pattern. This indicates a reduced amount of probability in this pattern, which corresponds to the description of the system pattern in Chapter 7.1.5.

The same tendency, namely that the values for both trace lengths lie close upon each other, holds true for the results of Huselius' Sum of Divergence. Especially, those from trace recordings at system level have a noticeable small gap between them.

The amount of events resulting from simulating the model and the time it takes CoreTAna to process these events show, again, no significant difference in comparison to the previous system model.

In summary it can be said that CoreTAna yields comprehensible and useful results. It is shown that traces at process level allow one to generate only a rough behavioural description of a system because of the missing details and, thus, the length of the trace does not play a big role. But the more information CoreTAna gets as input, i.e.,

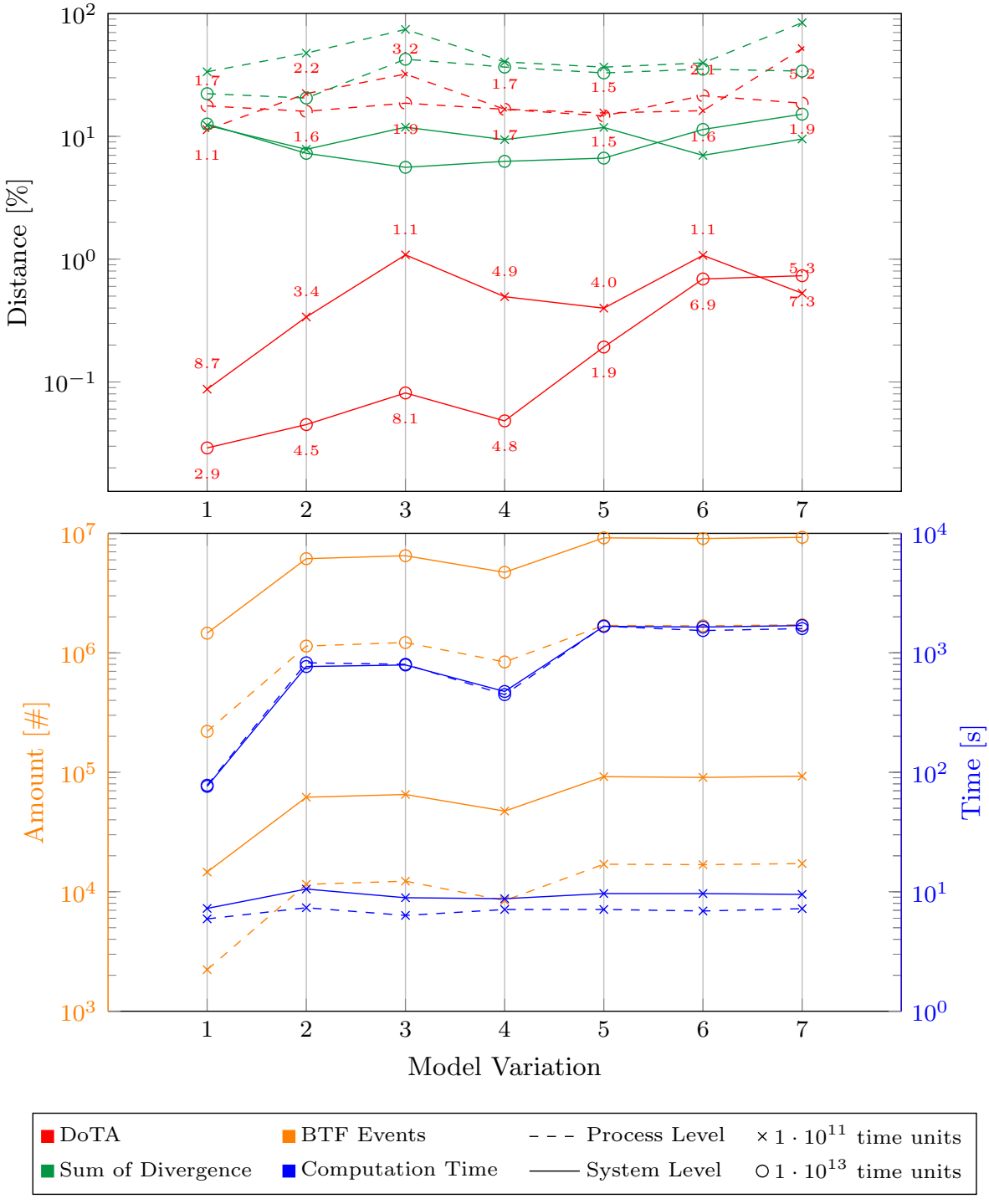


Figure 8.5.: Results of CoreTAna for the variation of system architecture pattern 'State Machine Feedback Loop'. Coloured marks and line styles are used as in Figs. 8.2, 8.3 & 8.4.

events at system level and recordings that cover a longer period of time, the more precise is the resulting model.

Our measure DoTA represents a crucial part in this evaluation by determining the results in a comprehensible way. This means that the resulting values do not tend to vary. Thus, small changes in the complexity always show only small difference and missing information has a big impact throughout the entire evaluation. In contrast, such a non-volatile behaviour is not shown by Huselius' Sum of Divergence. Although the results of the measure are nearly congruent with those of our measure for trace recordings at process level, they sometimes vary massively at system level, which makes the measure inconsistent.

The time it takes CoreTAna to reverse engineer a model is also comprehensible. There is not much difference between the resulting times of trace recordings at process and system level. Basically, they are congruent because the main effort is spent on processing the events in chronological order and analysing the effects, finally, takes only a minor amount of time. Besides that, we see that the time increases roughly linearly with the amount of time covered by a trace recording instead of the included amount of events.

With this knowledge, we apply CoreTAna to randomly generated systems to explore its performance in the broad design space of real-time systems.

Table 8.5.: Detailed Results of CoreTAna for 'State Machine Feedback Loop'.

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Distance of Timed Actions [%]	1	11.315112	17.685999	0.0873491	0.0290390
	2	22.267737	15.970018	0.3381904	0.0449968
	3	32.124778	18.620338	1.0841904	0.0814129
	4	16.613458	16.599144	0.4946097	0.0482207
	5	15.497905	14.658455	0.3988198	0.1925746
	6	16.169657	21.441146	1.0742245	0.6898238
	7	51.823490	18.602887	0.5277344	0.7329283

	Variation	Level of Detail and Length of Trace [time units]			
		Process Level		System Level	
		10^{11}	10^{13}	10^{11}	10^{13}
Sum of Divergence [%]	1	33.534720	22.249503	12.277420	12.597820
	2	47.545493	20.493800	7.8316856	7.2718625
	3	74.188149	42.445496	11.864354	5.5936270
	4	40.329721	36.654553	9.3963209	6.2592251
	5	36.596220	32.822951	11.811192	6.6458784
	6	39.612082	35.236438	7.0037333	11.376730
	7	84.183426	33.965710	9.5088655	15.147561
Amount of BTF Events [#]	1	2229	220029	14625	1460026
	2	11536	1138420	61870	6133606
	3	12262	1219472	65184	6497826
	4	8435	838957	47310	4717998
	5	17003	1694568	91876	9169264
	6	16863	1683061	90581	9053512
	7	17209	1715591	92700	9254323
Computation Time [s]	1	5.9284746	77.912520	7.2566530	76.278453
	2	7.3526964	824.84762	10.531692	767.76222
	3	6.3379423	801.60637	8.9131484	791.39058
	4	7.1040692	446.73755	8.7495989	474.62437
	5	7.1094310	1669.8961	9.6598348	1665.9249
	6	6.9041547	1535.6398	9.6521396	1642.6185
	7	7.2183878	1594.9064	9.5066149	1691.1236

8.2. Randomly Generated Systems

The randomly generated systems are not completely random, but their generation follows configurable settings. This allows us to influence the possible design space such that a desired mixture of system architectures can be ensured. To do so, we

analysed the systems of our customer projects and set the configuration in such a way that we get system models with a realistic amount of processes and runnables and reasonable internal behaviour.

Altogether, we generate a total of 1000 different systems. The number of tasks in each system is chosen randomly between 9 and 20. All tasks are activated harmonically by periodic alarms, with an offset between 0 and 50 ms and a period between 1 and 1000 ms. Their execution times are determined based on both their load, which can be between 1 and 20 %, and the number of called runnables, which each cover between 30,000 and 100,000 instructions. This results in roughly 5 to 100 runnables per task that are bundled in 6 to 13 coherent call sequences. Conditional statements are added to each task in such a way that the execution of any number of these call sequences depends on random values within the domain of one of the 1 to 20 generated data signals. This is repeated with the budgeted call sequences, in order to enable nested control flows. Data dependency is then established by adding, with a probability of 50 %, a signal access to each runnable, which writes a random value within the domain of that data signal.

A purely periodic task, which is roughly defined as a task without any conditional statements, is consequently generated with an approximate probability of at least $\frac{1}{9}$ (or $11.\overline{1}$ %). This is due to the fact that roughly 5 to 100 runnables per task are bundled into 6 to 13 coherent call sequences, which yield 1 to 9 call sequences per task. Consequently, the probability that a task representing a server in a 'Client-server without Reply' architecture is generated, is equal to the probability that one conditional statement is present, which is at least $\frac{1}{9}$, and the fact that there is no write access to that data signal within the task. Since 11 data signals are generated on average per system, this results in a probability of at best $\frac{1}{99}$ or $1.\overline{01}$ %. In contrast, a task representing a state machine has to modify the data signal on which it depends. Because a runnable has write access to each signal in 50 % of the cases, the resulting probability is roughly 5 %.

Fig. 8.6 visualises how capable CoreTAna is to reverse engineer the randomly generated models, which is listed in detail in Tab. 8.6. To do so we used both trace recordings that cover the system's internal behaviour at process level and traces at system level. The stacked bar chart at the bottom depicts that 96 % of the runnables differ by less than 3 % (sum of white and light red bars). This means that the internal behaviour of nearly all runnables can be reverse engineered very precisely. However, there are also scattered outliers, which indicate differences in the runnable behaviour

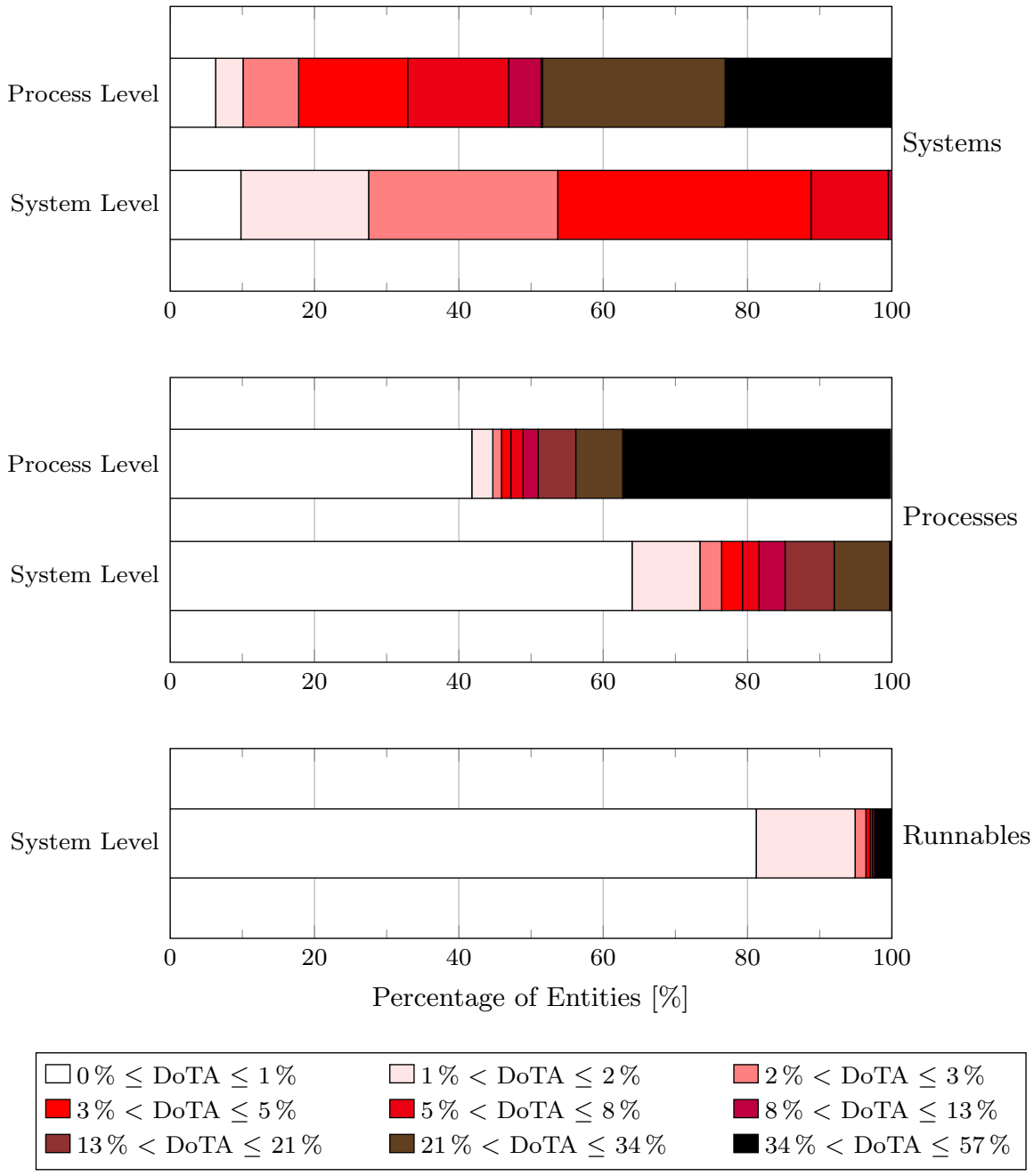


Figure 8.6.: Results from Randomly Generated Systems. Stacked bar chart of CoreTAna's results for reversely engineering the randomly generated models, each based on a trace recording that covers 1 s of the system's execution. The coloured bars mark the percentage of systems, processes, and runnables in which their reversely engineered behaviour differs from the original one by a certain distance value according our distance measure DoTA. Adapted from [13].

as high as 57 % (black bar).

Due to the fact that multiple runnables are mapped to a single process, the process's difference is added up by the difference of each runnable called, which consequently yields worse results. Nevertheless, 64 % of the processes differ by less than 1 % (white bar) according or distance measure DoTA. Using trace recordings at process level for reverse engineering visualises a completely different result. Only 42 % of all processes of the randomly generated systems show a difference of 1 % or less, which is nearly as much as the 37 % of processes that differ by more than 34 % (black bar).

The fact that all resulting differences for the overall systems are lower than 10 %, shown in the stacked bar chart by the absence of accordingly coloured bars, demonstrates that CoreTAna performs well in capturing a system's dynamic behaviour when trace recordings at system level are used as input. However, if a trace at process level is used, also the CoreTAna's performance is very limited. The results show that in only 50 % of the cases the temporal behaviour described by the reserve engineered model differs by less than 10 % from the general behaviour of the original real-time software.

To investigate the effect of this fact on real-life problems, we show the use of CoreTAna in actual industrial projects.

Table 8.6.: Detailed Results of Randomly Generated Systems. Amount of system, process and runnable entities from randomly generated traces at process and system level that have at least a difference less than or equal to a specific DoTA value.

DoTA [%]	Amount of Entities [%]				
	Systems		Processes		Runnables
	Process Level	System Level	Process Level	System Level	System Level
1	9.8	6.3	64.01	41.79	81.18
2	27.5	10.1	73.46	44.68	94.84
3	53.7	17.8	76.49	45.86	96.36
4	75.6	26.3	78.17	46.56	96.73
5	88.8	32.9	79.37	47.19	96.92
6	95.8	38.1	80.22	47.89	97.05
7	98.2	43.1	80.96	48.40	97.14

DoTA [%]	Amount of Entities [%]				
	Systems [%]		Processes [%]		Runnables [%]
	System Level	Process Level	System Level	Process Level	System Level
8	99.5	46.9	81.71	48.92	97.21
9	99.8	49.1	82.29	49.24	97.28
10	100	50.5	82.90	49.58	97.34
11	100	51.0	83.46	49.94	97.39
12	100	51.4	84.20	50.45	97.44
13	100	51.4	85.26	51.07	97.47
14	100	51.4	86.40	51.86	97.50
15	100	51.4	87.21	52.55	97.53
16	100	51.4	89.74	54.86	97.56
17	100	51.4	90.25	55.19	97.57
18	100	51.5	90.66	55.50	97.60
19	100	51.5	91.15	55.77	97.62
20	100	51.5	91.55	56.07	97.64
21	100	51.6	92.04	56.38	97.66
22	100	51.8	92.71	56.77	97.69
23	100	52.2	93.64	57.79	97.72
24	100	52.9	95.31	59.22	97.73
25	100	53.7	97.86	61.33	97.74
26	100	54.2	98.78	62.07	97.74
27	100	55.8	98.99	62.13	97.74
28	100	56.8	99.21	62.17	97.74
29	100	58.3	99.33	62.32	97.74
30	100	60.5	99.45	62.54	97.74
31	100	64.5	99.50	62.64	97.74
32	100	68.0	99.56	62.74	97.74

DoTA [%]	Amount of Entities [%]				
	Systems [%]		Processes [%]		Runnables [%]
	System Level	Process Level	System Level	Process Level	System Level
33	100	71.9	99.66	62.84	97.74
34	100	76.9	99.73	62.93	97.74
35	100	81.0	99.77	63.01	97.74
36	100	86.2	99.81	63.14	97.74
37	100	90.3	99.86	63.30	97.74
38	100	93.0	99.91	63.72	97.74
39	100	95.7	99.93	64.22	97.74
40	100	97.7	99.94	64.37	97.74
41	100	99.2	99.95	92.83	98.10
42	100	99.6	99.96	94.29	98.10
43	100	100	99.96	95.18	98.10
44	100	100	99.96	96.81	98.10
45	100	100	99.96	97.08	98.10
46	100	100	99.96	97.49	98.10
47	100	100	99.96	98.01	98.10
48	100	100	99.97	99.72	98.10
49	100	100	99.99	99.90	98.10
50	100	100	99.99	99.90	98.10
51	100	100	99.99	99.90	98.10
52	100	100	100	99.95	98.10
53	100	100	100	99.98	98.10
54	100	100	100	99.99	98.11
55	100	100	100	99.99	98.11
56	100	100	100	99.99	98.12
57	100	100	100	99.99	98.35

DoTA [%]	Amount of Entities [%]				
	Systems [%]		Processes [%]		Runnables [%]
	System Level	Process Level	System Level	Process Level	System Level
58 – 100	100	100	100	100	100

8.3. Industrial Case Studies

In addition to the synthetic benchmark and the randomly generated systems, four industrial case studies are conducted. Because TA is a tool vendor and consulting company whose customers are mainly automotive OEMs and Tier-1s, three of these case studies illustrate the use of CoreTAna in the automotive domain. However, to show that the developed algorithms are not only limited to the automotive domain and that they are usable for real-time systems in general, we also apply CoreTAna to trace recordings from a former customer project in the telecommunication domain.

8.3.1. Automotive Case Studies

The case studies are chosen in such a way that the reverse engineering copes in each case study with a trace recording that stores information about the system at a specific level of detail. At first, we apply CoreTAna to a trace recording of an engine management system, which presents the hardest challenges regarding real-time in the automotive development. Because of the precise timing that is required for the complex processes in a combustion engine such as injection, it is technically impossible to trace more details than the interaction of tasks. A steering system, which is used for the next case study, is less complex and allows one to apply function trace, i.e., all process and runnable events be observed without any temporal impact on its execution. Because actual industrial systems are too complex which makes it technically impossible to trace all details of its internal behaviour including data accesses, we use trace recordings generated when simulating the model of the FMTV Challenge in the third case study.

We evaluate CoreTAna's performance and the achieved quality of the reverse engineering by us using our proposed measure Distance of Timed Actions (DoTA); the results are visualised in Fig. 8.7.

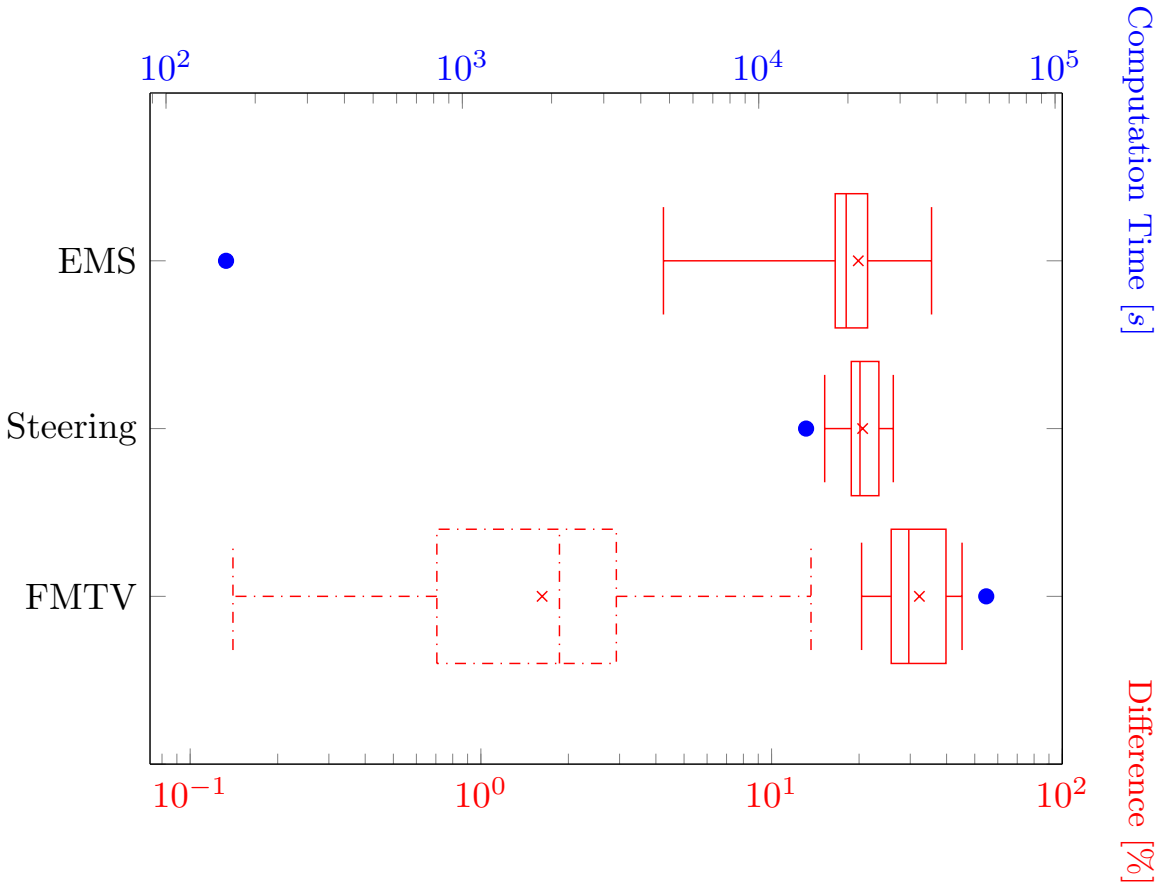


Figure 8.7.: Results of Automotive Case Studies. Box plots with whiskers from minimum to maximum, summarising the differences between each task/ISR in the individual industrial case studies according our distance measure DoTA. A cross marks the result of the difference measure for the overall system. The dashed box plot visualises the results of the FMTV Challenge with the given hardware model. Reprinted from [13].

8.3.1.1. Engine Management System (EMS)

In a first industrial project, an engine management system (EMS) consisting of 75 tasks is analysed. To do so, a trace covering 6.5 s of the system's execution was provided in a 40 MB file containing $90 \cdot 10^3$ events. Because of the high complexity of the system under investigation and technical limitations, it was only possible to observe and record task state transitions. Thus, no detailed insight into the system's behaviour can be inferred.

The visualisation in Fig. 8.7 of the differences between each task, which are listed in detail in Tab. 8.7, shows that the quality of CoreTAna's reverse engineering is inconsistent. Some tasks contain only a small amount of variability, which is why their behaviour is reflected quite well with a difference of only 4 %. Other tasks, however,

show a difference of 35 %, which indicates a lack of detail. Nevertheless, the overall difference of 19.88 % is in line with the results of the synthetic models in Chapter 8.1, where evaluations of CoreTAna's capability to handle the limited accuracy in logged data yielded differences between 13 % and 52 %.

In addition, the time it takes CoreTAna to generate the model is stated in the figure. The blue dot shows that the $90 \cdot 10^3$ events in the trace recording are processed in 159 s, resp., 2.65 min, which is roughly 3 to 4 times longer than analysing an according trace of the synthetic models.

Table 8.7.: Detailed Results of Case Study 'Engine Management System (EMS)'.

Process	Difference (DoTA) [%]	Computation Time [s]	Process	Difference (DoTA) [%]	Computation Time [s]
P1	27.26822	0.75112	P39	16.25324	0.084928
P2	29.33211	0.096376	P40	15.50078	0.076498
P3	17.56982	0.103602	P41	16.12675	0.083124
P4	18.84569	0.10029	P42	17.95330	0.073786
P5	23.45587	0.097278	P43	15.68794	0.08764
P6	27.41339	0.088846	P44	21.07741	0.071076
P7	24.05519	0.075294	P45	20.55802	0.080414
P8	18.48474	0.073486	P46	16.88526	0.063846
P9	28.18625	0.07469	P47	15.94663	0.063246
P10	15.57548	0.071078	P48	16.26116	0.070172
P11	17.40405	0.075896	P49	30.37618	0.066258
P12	37.42805	0.076194	P50	16.86565	0.069572
P13	25.62245	0.058428	P51	17.78637	0.068668
P14	20.10103	0.068064	P52	16.49458	0.06927
P15	15.54986	0.068064	P53	16.77918	0.068666
P16	23.90535	0.06144	P54	16.68903	0.070776
P17	17.90769	0.065354	P55	15.80121	0.068668
P18	20.16454	0.067462	P56	19.68019	0.068366
P19	17.25456	0.069572	P57	15.74979	0.080714

Process	Difference (DoTA) [%]	Computation Time [s]	Process	Difference (DoTA) [%]	Computation Time [s]
P20	16.56956	0.0067766	P58	24.44865	0.059934
P21	16.98214	0.067464	P59	29.57777	0.056318
P22	19.99930	0.081918	P60	28.17908	0.057524
P23	22.17637	0.062946	P61	17.98144	0.065354
P24	15.86590	0.065054	P62	16.67649	0.071982
P25	17.58791	0.073184	P63	31.87213	0.055416
P26	16.46475	0.087338	P64	28.11581	0.060838
P27	17.36096	0.067162	P65	18.68762	0.070172
P28	18.45034	0.069872	P66	24.61532	0.061438
P29	18.44861	0.080714	P67	18.42596	0.063548
P30	18.20959	0.073786	P68	21.21538	0.065052
P31	19.91532	0.072882	P69	19.74189	0.067464
P32	18.98483	0.089448	P70	15.89531	0.073486
P33	22.49819	0.065656	P71	16.69517	0.071074
P34	35.40044	0.067464	P72	18.53464	0.072584
P35	24.49537	0.067762	P73	16.41568	0.070172
P36	4.19010	0.091858	P74	16.55767	0.07138
P37	28.29331	0.063848	P75	18.26587	0.068968
P38	19.02603	0.07198			

8.3.1.2. Steering System

For this industrial case study, a model of a steering system software is generated. The system under investigation consists of 18 tasks and interrupt service routines. Altogether, these call 130 different runnables. The employed hardware platform is a dual core processor with a frequency of 120 MHz for each processing unit. The starting point of the reverse engineering is a trace recording that covers 30 s and contains all task and runnable calls performed during that time. This resulted in roughly $27 \cdot 10^6$ events and a file of 1.7 GB, i.e., the system produces roughly one million events per

second.

In contrast to the previous industrial case study, where only task events have been recorded, the details added by also observing function calls lead to less spread. The difference for each task lies just between 14.93 % and 26.23 % as shown in Fig. 8.7. However, the overall difference of 15.54 % is slightly higher than in the previous case study, which is probably due to the trace containing more information. Nevertheless, the trace is missing knowledge about data accesses, so that the varying internal behaviour due to data dependencies cannot be determined in full detail.

Although the trace recording used in this case study contains 300 times the amount of events than that in the previous one, CoreTAna completes the reverse engineering in around 14,450 s, resp., 4 h. This corresponds to the computation times that are determined by the synthetic benchmark.

8.3.1.3. FMTV Challenge 2016

This industrial case study is inspired by the *Formal Methods for Timing Verification* (FMTV) Challenge 2016 [68], where a model of an industrial real-time system has been published in order to discuss solutions to concrete timing verification problems. Although no trace recordings from the actual system are provided, we have used this model and the TA Simulator [70] to generate a simulation trace. Because this model-based timing simulation is a commercial tool that is employed by many Tier-1s and OEMs in the automotive industry and because the provided model is very detailed, it is reasonable to assume that the generated trace recording corresponds very closely to the actual system behaviour.

The FMTV Challenge's model describes a full-blown engine management software that consists of 10 periodic tasks and 11 ISRs that interact with the system sporadically. The functionality is provided by 1250 runnables, and roughly 10,000 different data signals are accessed for communication. On the hardware side, a microcontroller architecture with four symmetric cores is available for processing. Each core has access to its local RAM and to the shared global RAM via a crossbar. Based on this model, the system's execution has been simulated for 30 s. During that time, roughly $341 \cdot 10^6$ events have occurred and been recorded in a trace, which resulted in a file size of roughly 17 GB.

Fig. 8.7 visualises the resulting differences between each task and ISR, which are listed in detail in Tab. 8.8. Although the trace contains all details of the system behaviour, the results are still around 30 %. Motivated by this rather disappointing

outcome, a closer examination showed that the hardware limitations of the crossbar together with cumulative data accesses caused tasks to wait. Because the correct modelling of the hardware properties are not subject of our reverse engineering, we have repeated the reconstruction with the hardware model as given input, so as to evaluate CoreTAna's performance regarding how closely the software model reflects the actual behaviour. The results are shown in Fig. 8.7 as a dashed box plot. With the differences mainly being around 1 %, these results correspond more to the expected outcome by being in line with our synthetic benchmark.

Noticeable in the figure is also the time it takes CoreTAna to generate a model from the trace recording, which is again marked by a blue dot. Although the amount of events in the trace recording increased by a factor of 13 compared to that in the previous case study, the resulting computation time of 58,584 s, resp., 16.3 h is only 4 times longer. The reason for this is that the trace recordings that contain a high level of details and that cover a long period of time reduce the amount of time that CoreTAna has to spend on analysing the effects after all events have been processed by adding more information to the decision process. This corresponds to the results of the synthetic benchmark in Chapter 8.1 where processing a trace recording at system level takes nearly the same amount of time than analysing one at process level.

Table 8.8.: Detailed Results of Case Study 'FMTV Challenge 2016'.

Process	Difference (DoTA) [%]		Computation Time [ms]	
	Deduced HW	Given HW	Deduced HW	Given HW
Angle_Sync	39.7789340124	0.3384393561	0.192448	0.106616
ISR_1	30.7453931113	6.8977576547	0.052102	0.055416
ISR_10	23.6108101193	1.6424759367	0.052104	0.058124
ISR_11	25.5212024697	2.0535186001	0.055416	0.056922
ISR_2	27.6081764748	1.5509465746	0.071078	0.052404
ISR_3	28.5702782510	5.8641149974	0.065658	0.06686
ISR_4	27.2891897052	1.6163047883	0.058426	0.061138
ISR_5	23.5894187879	0.2756244249	0.057222	0.06144
ISR_6	27.6144606314	1.1682904454	0.044574	0.046982
ISR_7	28.3381694067	2.1841490585	0.046982	0.05361
ISR_8	24.1850153803	1.3252726897	0.06415	0.066258

Process	Difference (DoTA) [%]		Computation Time [ms]	
	Deduced HW	Given HW	Deduced HW	Given HW
ISR_9	21.5937426961	5.7907877851	0.066258	0.075594
Task_1000ms	45.3900874549	13.8088965717	0.057824	0.052404
Task_100ms	40.3456555802	1.7714186938	0.057524	0.057524
Task_10ms	40.2935588815	0.8648575138	0.13643	0.118962
Task_1ms	36.8073702601	0.2383956533	0.054512	0.054812
Task_200ms	43.0179164207	13.0182601801	0.442422	0.024778858
Task_20ms	40.3033767057	2.4341276595	0.106916	0.107518
Task_2ms	34.2606232091	0.5635644907	0.060836	0.059932
Task_50ms	39.3525579074	2.9851188616	0.10842	0.102396
Task_5ms	34.1450933199	2.8254274324	0.052402	0.05391

8.3.2. Further Case Study in Telecommunication

This industrial case study describes CoreTAna's use in the context of an evaluation project together with a customer from the telecommunication domain. The purpose of this project was to collect requirements regarding the modelling and simulation of real-time systems that are specific to the telecommunication domain in order to estimate the effort required to expand into that domain.

The topic of this project was the analysis of the software for a Long-Term Evolution (LTE) chipset like it is used in mobile phones. Although this system deals also with meeting hard real-time requirements, the design of the overall system is totally different. The system consists of in total 127 ISRs and tasks. However, only 28 of those are activated periodically. The other processes are chained which means the periodically activated processes trigger them via inter-process activations they, again, trigger other processes. To get different activation patterns that way, processes are not activated every time but only every n-th time by the inter-process activation.

The employed hardware platform is a dual core processor with a frequency of 543 MHz for each processing unit. The starting point of the reverse engineering is a trace recording that covers 30 s of the system's dynamic behaviour at process level, i.e., it contains all process events performed by the system during that time. This resulted in roughly $5.3 \cdot 10^6$ events and a file of 330 MB.

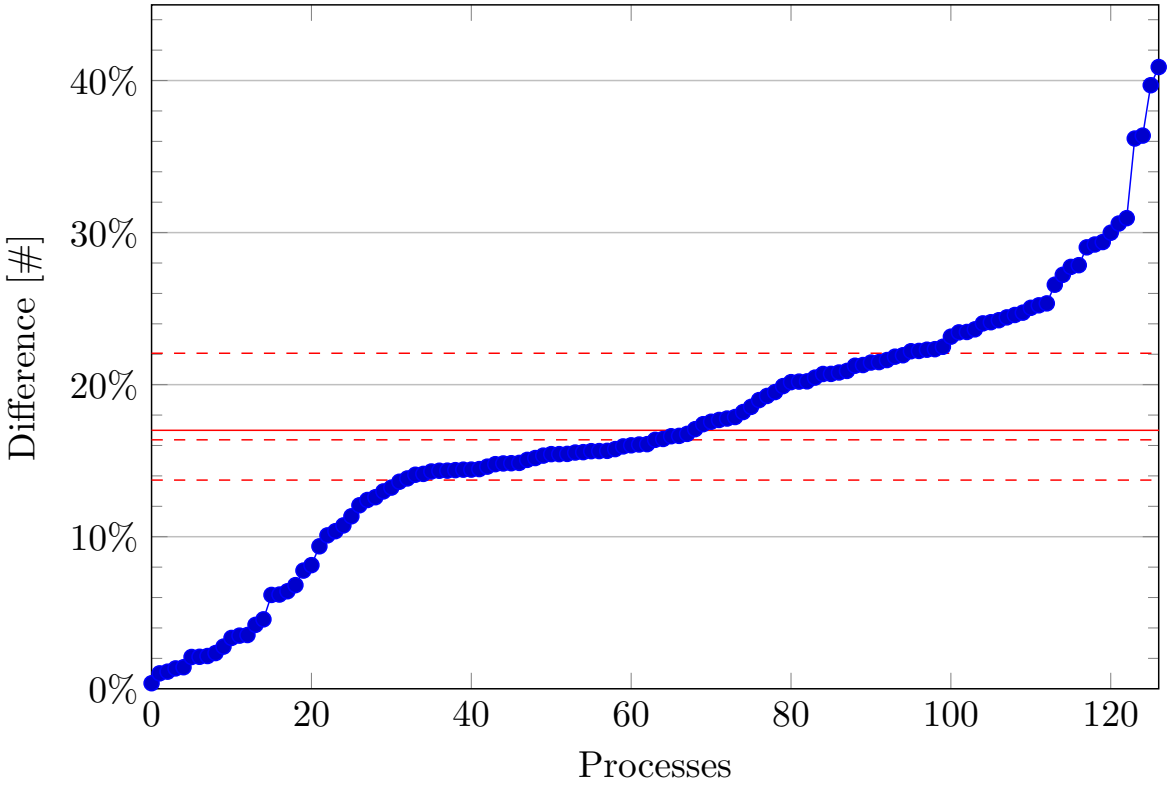


Figure 8.8.: Results of Telecommunication Case Study. Line chart showing in blue the differences between each task/ISR in the telecommunication case study according our distance measure DoTA. The dashed red lines mark the lower and upper quartile and the median of the results and the solid red line marks the result of the difference measure for the overall system.

The resulting differences between each task and ISR from the original trace recording and one generated when simulating CoreTAna's reversely engineered model, which takes CoreTAna roughly 283.7 s, resp., 4.7 min to generate, are listed in detail in Tab. 8.9 and are visualised in Fig. 8.8. There, the differences are ordered from smallest to largest and go from 0.3 % up to 40.9 %. Thus, CoreTAna can reverse engineer the dynamic behaviour of some processes quite accurately. Because the aforementioned design in which processes are not activated every time but only every n-th time by the inter-process activation cannot be reproduced in the AUTOSAR-compliant model, there are also processes that do not behave fairly similar. This modelling deficit leads to a significant spread of the results. Other than that, the 17 % difference of the overall system and the location of the quartiles (13.7 % and 22 % vs. 17 % and 21 % in EMS) are in line with those of the automotive case study Engine Management System (EMS).

Table 8.9.: Detailed Results of Case Study 'Telecommunication'.

Process	Difference (DoTA) [%]	Process	Difference (DoTA) [%]	Process	Difference (DoTA) [%]
P1	17.56	P44	15.63	P87	15.76
P2	6.82	P45	20.80	P88	7.77
P3	2.35	P46	15.43	P89	15.33
P4	16.37	P47	24.74	P90	15.94
P5	21.45	P48	29.38	P91	15.63
P6	14.41	P49	25.22	P92	21.93
P7	18.54	P50	14.08	P93	20.46
P8	21.29	P51	14.85	P94	13.62
P9	23.64	P52	19.90	P95	4.20
P10	24.58	P53	17.07	P96	24.24
P11	17.40	P54	15.56	P97	20.70
P12	39.69	P55	23.46	P98	0.36
P13	15.17	P56	21.24	P99	1.12
P14	20.90	P57	12.41	P100	30.60
P15	23.16	P58	24.11	P101	36.19
P16	16.61	P59	25.34	P102	17.68
P17	16.76	P60	21.48	P103	16.09
P18	6.17	P61	16.43	P104	16.63
P19	3.53	P62	21.62	P105	4.57
P20	1.42	P63	2.77	P106	2.09
P21	15.45	P64	12.60	P107	2.15
P22	20.20	P65	24.03	P108	2.10
P23	18.20	P66	30.95	P109	9.37
P24	20.16	P67	1.01	P110	3.34
P25	40.89	P68	12.07	P111	24.42
P26	16.06	P69	13.83	P112	27.22

Process	Difference (DoTA) [%]	Process	Difference (DoTA) [%]	Process	Difference (DoTA) [%]
P27	21.84	P70	19.26	P113	19.51
P28	22.30	P71	14.83	P114	20.71
P29	29.22	P72	20.22	P115	10.09
P30	22.49	P73	18.98	P116	17.87
P31	14.82	P74	14.41	P117	16.01
P32	11.34	P75	14.38	P118	27.86
P33	6.41	P76	23.44	P119	36.38
P34	26.57	P77	14.34	P120	6.20
P35	14.77	P78	25.05	P121	22.20
P36	14.45	P79	10.74	P122	29.03
P37	15.54	P80	10.36	P123	22.33
P38	14.61	P81	29.99	P124	15.43
P39	27.74	P82	14.29	P125	22.22
P40	14.34	P83	15.65	P126	14.13
P41	12.98	P84	3.50	P127	13.23
P42	15.05	P85	8.13		
P43	17.77	P86	1.34		

8.3.3. Summary

The industrial case studies show that CoreTAna is not only capable of handling complex and series systems but also that it can reverse engineer a model of such a system with an acceptable quality and in a reasonable amount of time. Although trace recordings at process level allow one only to get a rough description of a system's internal behaviour because of the missing level of detail in the trace, it constitutes the starting point of the reverse engineering process by giving an overview of the system. Based on the resulting difference of a process calculated by our measure DoTA, the model can by refined afterwards, e.g., by using a trace recording of a specific process at system level as input. Although it is technically not possible to record the internal

behaviour of an entire industrial system at the moment, the case study of the FMTV challenge proves that CoreTAna yields a very good representation of the system from trace recording with the necessary level of detail.

The conclusions of this chapter show that the content of CoreTAna's input plays a crucial part for how well a synthesised model reflects the timing behaviour of the original system, which is why we have a closer look at the quality of trace recordings next.

8.4. Reasoning on the Quality of a Trace Recording for Reverse Engineering

The general drawback of reverse engineering is that only as much information can be deduced as is available in the input. This means in our case that it is not possible to reproduce the internal behaviour of tasks from trace recordings at process level because of the missing runnable and data signal events. For this reason, our goal is to put as much information about the system's runtime behaviour in a trace recording as possible.

One possibility to do so is to increase the period of time that is covered by a trace recording as shown in Tab. 8.10 and visualised in Fig. 8.9. There, trace recordings of the FMTV challenge model are employed to CoreTAna and compared to the traces

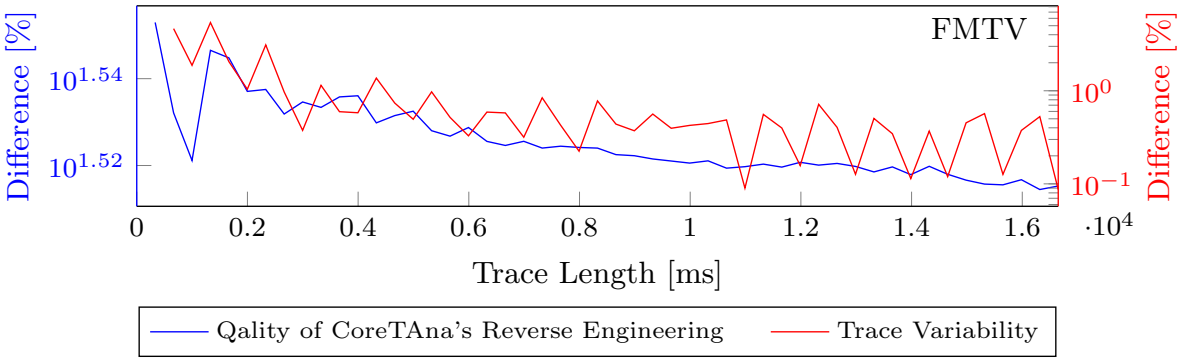


Figure 8.9.: Comparison of Trace Variation and Quality of CoreTAna's Reverse Engineering for the 'FMTV Challenge 2016' Model. The blue line denotes the resulting quality of CoreTAna's reverse engineering with increasing trace length, i.e., the result of our measure Distance of Timed Actions (DoTA) for comparing a trace of the pattern with one generated when simulating CoreTAna's reversely engineered model. The red line shows how the information content within the trace changes over time according DoTA.

generated when simulating the synthesised model using our Distance of Timed Actions (DoTA) measure. With each trace the period of time that is covered is increased by 333 ms ($\frac{1}{1000}$). The resulting values are depicted in blue in Fig. 8.9 and show that the quality of the reverse engineering tends to converge towards a limit. This means that from a certain point on recording for an even longer period does not add further information for reverse engineering. Thus, the big challenge is to determine this point in the trace or, alternatively, to make a statement about the quality of the synthesised model that can be expected from a given trace recording.

Table 8.10.: Detailed Results of Comparison of Trace Variation and Quality of CoreTAna's Reverse Engineering for the 'FMTV Challenge 2016' Model.

Length [ms]	Quality (DoTA) [%]	Variation (DoTA) [%]	Length [ms]	Quality (DoTA) [%]	Variation (DoTA) [%]
333	35.750679		8658	33.329179	0.438585
666	34.076924	4.648972	8991	33.311732	0.371765
999	33.228809	1.876520	9324	33.256415	0.562073
1332	35.228333	5.405618	9657	33.219371	0.395579
1665	35.088200	2.056216	9990	33.180795	0.424671
1998	34.469732	1.038171	10323	33.221688	0.444274
2331	34.501293	3.108156	10656	33.093740	0.486188
2664	34.057396	0.970886	10989	33.117108	0.089936
2997	34.272702	0.375603	11322	33.165323	0.557543
3330	34.175360	1.143149	11655	33.111601	0.398459
3663	34.367096	0.596195	11988	33.194851	0.157094
3996	34.387209	0.580546	12321	33.144646	0.714682
4329	33.896480	1.368369	12654	33.175625	0.404691
4662	34.028180	0.737704	12987	33.122849	0.126754
4995	34.108547	0.494351	13320	33.024830	0.504356
5328	33.756647	0.973412	13653	33.115016	0.345943
5661	33.658834	0.517732	13986	32.980784	0.114001
5994	33.812701	0.328096	14319	33.123689	0.368636

Length [ms]	Quality (DoTA) [%]	Variation (DoTA) [%]	Length [ms]	Quality (DoTA) [%]	Variation (DoTA) [%]
6327	33.566830	0.590101	14652	32.987357	0.119663
6660	33.494483	0.578175	14985	32.882363	0.452551
6993	33.566716	0.317032	15318	32.814908	0.568291
7326	33.449075	0.840793	15651	32.801490	0.126521
7659	33.481178	0.426744	15984	32.888858	0.375151
7992	33.456345	0.224319	16317	32.720938	0.527586
8325	33.448162	0.778120	16650	32.780569	0.086768

Huselius [9] tackles this challenge not explicitly but defines a step called 'Resolution Analysis' in his reverse engineering approach, which determines "whether recorded data is sufficient to capture a model of the implementation" [9, p. 63]. This step analyses the set of trace recordings used for model generation together with the resulting model, and determines how many observations of each event are required for making probabilistic choices in the model. Afterwards, the trace recordings intended for validation are checked against this determined threshold. If this check fails, new traces are recorded that cover a longer period of time until both, the trace recordings used for generation and validation contain the same amount of information. The big disadvantage of this approach is that it is performed after the reverse engineering which consumes a lot of time. Thus, the goal must be to make a statement about the quality of the reverse engineering solely based on the trace recordings.

A general way for achieving this is proposed by Hamou et al. [80], who define the entropy of a trace, i.e., the average amount of information contained in the trace. Various aspects of a trace recording such as the number of different events or repetitions and its nesting depth are analysed and used to summarise the complexity of the trace. Although the authors show that this approach allows one to identify parts of the trace that perform complex behaviour, the defined entropy measure does not take any temporal aspects into consideration, which is essential in our case.

Due to this lack of a suitable existing solution for reasoning on the quality of a trace recording for reverse engineering, we developed a way to do so based on the DoTA measure. The general idea behind this approach is to take into consideration how much a trace changes over time. In case of the FMTV challenge example, this means that the trace is not compared to a trace that is generated when simulating

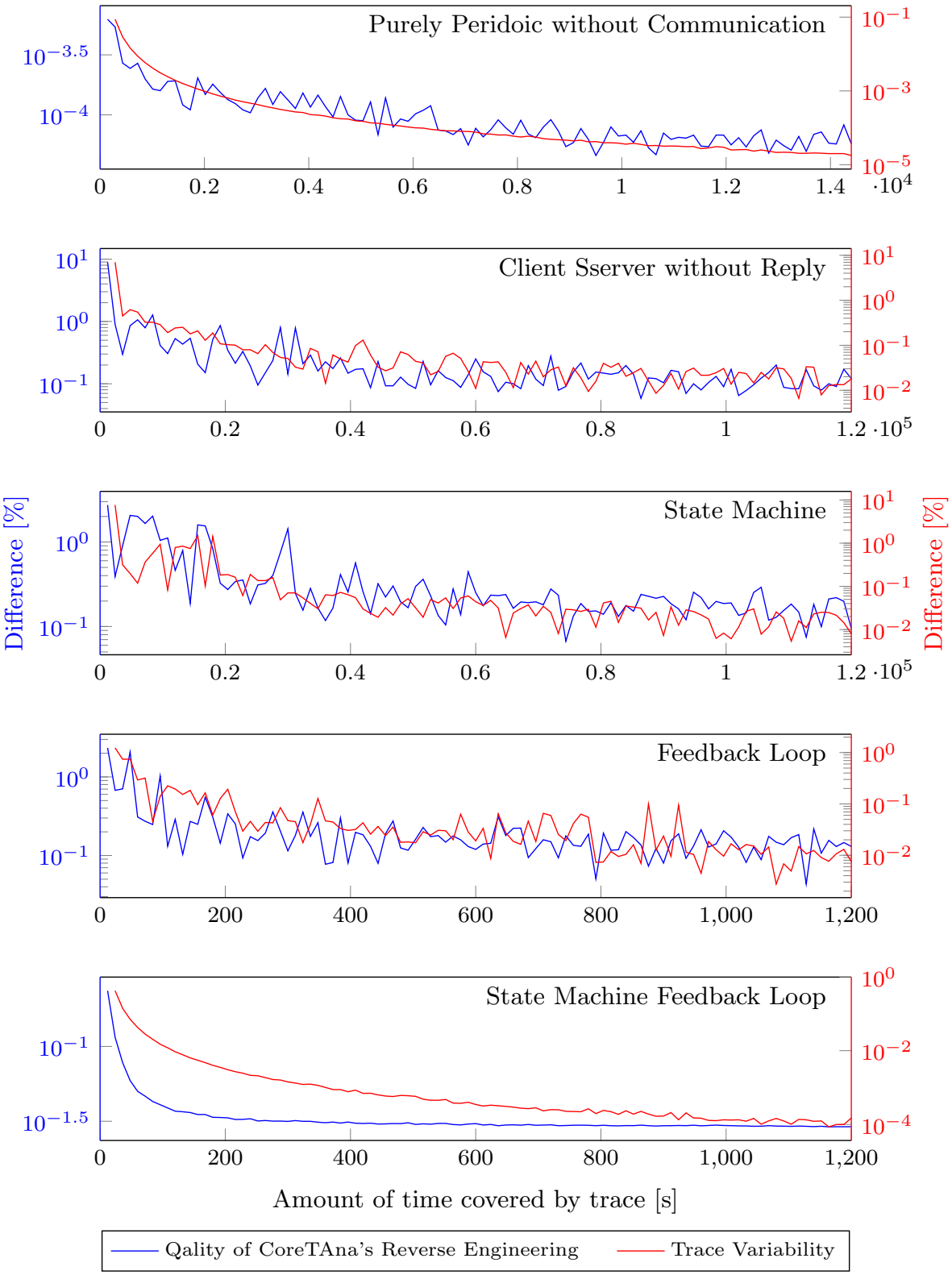


Figure 8.10.: Comparison of Trace Variation and Quality of CoreTAna's Reverse Engineering. The blue line denotes the resulting quality of CoreTAna's reverse engineering with increasing trace length, i.e., the result of our Distance of Timed Actions (DoTA) measure for comparing a trace of the pattern with one generated when simulating CoreTAna's reversely engineered model. The red line shows how the information content within the trace changes over time according to DoTA.

the synthesised model but to the one that covers 333 ms less of the system's temporal behaviour. Expectedly, the results which are added in red in Fig. 8.9 also converge towards a limit because at some point in time the trace recording contains already all possible situations that can be observed during the runtime of a system and, thus, its information content does not change any more.

To be able to generalise these observations, a larger pool of data is generated. Therefore, we use trace recordings from each initial model of the common architectural patterns as defined by the synthetic benchmark. The length of the trace recordings are chosen in such a way that they cover up to 400 hyper-periods of the system's internal behaviour. A trace at system level is employed to CoreTAna every four hyper-periods and the traces generated when simulating the synthesised model are compared with DoTA. Fig. 8.10 depicts these 100 measured values in blue along with the continuous changes of trace recordings which are shown in red.

It is noticeable in all the examples that both the red and blue lines converge nearly in equal measure. Just the amount of difference varies which indicates that there is

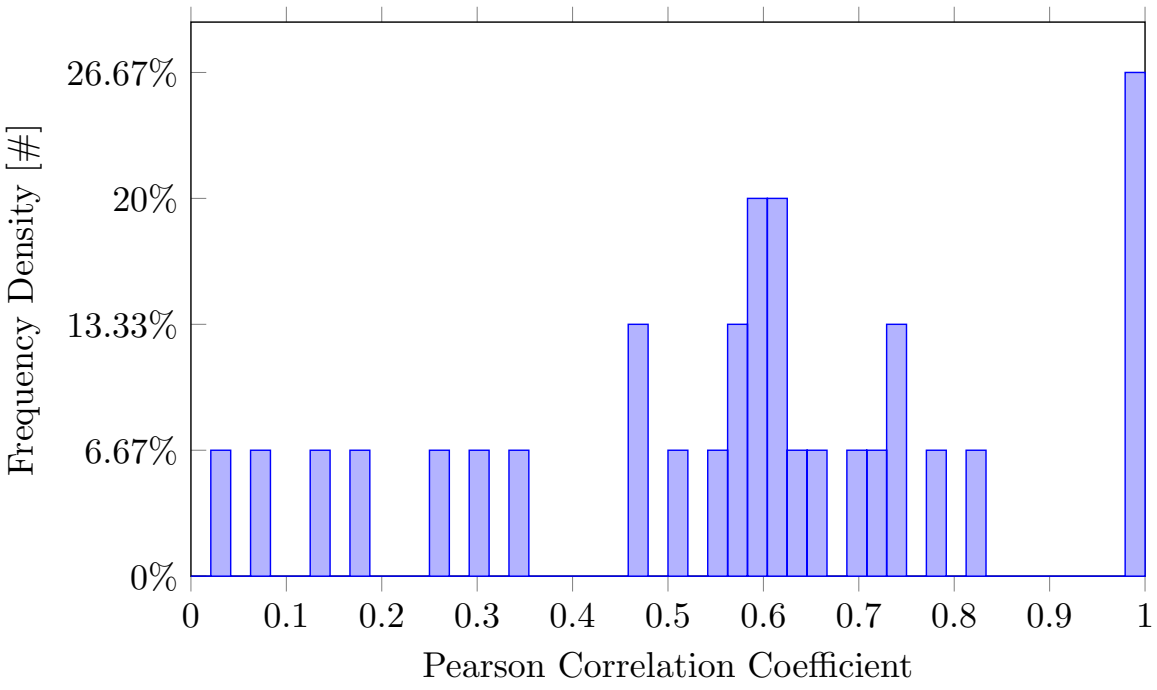


Figure 8.11.: Correlation of Trace Variation and Quality Variation of CoreTAna's Reverse Engineering. Histogram showing the distribution of results from determining the Pearson correlation coefficient between the continuous variation of a trace and the quality of CoreTAna's reverse engineering over time, for all model variations of the common architectural patterns as defined by our synthetic benchmark.

a linear correlation between the variation of a trace recording and CoreTAna's corresponding results. This means that if a trace changes, the results of the reverse engineering change alike. As a consequence, one can measure continuously the difference with our measure DoTA, which can be done at minimal expense even during trace recording. And once the trace recording shows no substantial change any more, also the results of the reverse engineering have reached a local convergence.

To support this theory, we measure the linear correlation between the resulting quality of CoreTAna's reverse engineering (blue) and the continuous variation of a trace (red) by determining the *Pearson correlation coefficient* [81]. Fig. 8.11 visualises the corresponding results for all model variations of our common architectural patterns. Despite a few outliers that have a Pearson correlation coefficient of less than 0.4, the histogram proves that there is a clear relationship between the two considered aspects. Hence, determining how much a trace changes over time with the help of our Distance of Timed Actions (DoTA) allows one to reason on the quality of a trace recording for reverse engineering.

Part III.

Summary

9

Conclusions and Outlook

This work investigated the reverse engineering of real-time system models from event trace recordings. We focused on the applicability to the automotive domain, for multiple reasons. First, model-based development has experienced gradual acceptance in the automotive domain over the last years because of the AUTOSAR standard [82]. This has resulted in the development of a variety of tools that work on the basis of models. Second, the current shift towards multi-core architectures, and the rapid rise of complexity that comes along with this shift, plays an essential part in contributing to the increased use of models during the development process. Thirdly, the research community has already developed a variety of solutions, but none is designed with the conditions of the automotive domain in mind and, thus, is not applicable there. For these reasons, we decided to pick up the considerations of an existing solution and to extend them in such a way that they are applicable to the automotive domain.

Experiences

Not only does the generation of an AUTOSAR-compliant artefact that can be used for further processing pose a major challenge. Also the fact that any reverse engineering approach must fit into the existing automotive development process. For example, the latest research in reverse engineering from trace recordings makes the assumption that as many traces as one desires can be produced if necessary, which would actually mean that the byte code or even the source code is available. This might be feasible in other domains, but the development in the automotive domain is distributed. The software architecture and the main functionality is developed by Tier-1 companies, but some functions of the system are also developed by the OEM itself, which means that the complete source code is not available at any given time. Furthermore, most related work assumes that the user spends much effort to achieve good results in the reverse engineering, e.g., by using a set of trace recordings that allows the system's behaviour to be analysed extensively. However, this is not the case in our situation.

The customer wishes to use model-based tools and benefit from their results, but it does not matter how these results are obtained. For this reason, creating an initial model of a system is not done by the OEM or Tier-1 but rather by the tool vendor itself, which means that each necessary information to do so has to be requested and, thus, must demand as little additional effort for the customer as possible.

In general, trace recording is considered improperly in existing solutions. It is, not only, infeasible to produce traces in any number, as mentioned above, but also that the system behaviour can be observed and recorded arbitrarily. The reason for assuming the opposite is because most related work considers software tracing. Although this technique allows one, in principle, to observe any part of the system, source code is required for doing so. More critical is the fact that each observation of a piece of information negatively impacts the timing behaviour. The current shift towards multi-core architectures makes the use of software tracing even worse, because parallelism generates even more information during execution. As a consequence, either software tracing is only applied to smaller scaled systems or just the start and stop of tasks are recorded in actual industrial applications. In contrast, hardware tracing does not alter a system's execution. However, it requires not only dedicated hardware but also deep knowledge in the applied tracing technique. From our experience, especially the latter in not available in many OEM and Tier-1 companies. This is due to the complexity of tracing and due to the fact that there is rarely a case of application in daily use that requires such a deep knowledge.

Results

In summary, this thesis presented two major scientific contributions in the area of reverse engineering in the automotive domain:

- CoreTAna: an automatic synthesis of an AUTOSAR-compliant model,
- DoTA: a measure for quantifying the difference between two trace recordings regarding the recorded timing behaviour.

These contributions include methods, implementations, and evaluations for our proposed reverse engineering approach, which is particularly tailored to the AUTOSAR methodology [82]. The main focus of this work is set on the industrial applicability of the contributions, i.e., assumptions that correspond to reality are considered in order to make the developed solution feasible. CoreTAna not only builds upon an existing approach, namely the one Huselius defined in [9], but is also built upon well established techniques such as Eclipse [83] and constraint programming [84]. Thus,

it fits seamlessly into the tool environment of the automotive development process in which such tools and techniques are in daily use. More important is the fact that the user gets additional confidence in the approach and its results in this way. Another achievement of CoreTAna is that it generates an AUTOSAR-compliant model, which paves the way for interoperability to a multitude of existing tools such as timing simulators or code generators because there is no need for specific adaptions. This has a major impact on companies such as newly emerged business ventures or ones from a different domain like aviation or automation because they can build upon this solution in order to get easier access to the automotive domain. In addition, the model is not closed or fixed to a special purpose but can be processed further and enriched with additional information.

CoreTAna and, especially, its implemented algorithms were assessed extensively in different ways. At first, they are evaluated on trace recordings from simulation models. Each model represents a common architectural pattern in the real-time software domain including feasible variations for the patterns. Besides this, CoreTAna has evaluated on simulation models that describe fictive but realistic systems with similar analysis challenges as observed in industrial systems. Finally, four industrial case studies illustrated CoreTAna's performance in actual customer projects. All three evaluation scenarios produced consistent results, which allow us to make the following general conclusions. In case trace recordings at system level are used as input and the system does not utilise any characteristics that are not supported by our reverse engineering approach, CoreTAna is able to create an AUTOSAR model that reflects the actual timing behaviour very precisely. The differences that still arise are so small that they do not matter to the intended use cases such as timing simulation or optimisation. The results turn out to be even more precisely if just the timing behaviour of individual runnables is considered and not that of the complete system or processes. By reducing the information in the trace to process level, the differences of the individual processes turned out to be either still very small or too high to use the results in a reasonable way. Nevertheless, a lot of valuable knowledge is acquired by CoreTAna, which can be used directly for documentation or as the foundation for model improvements with additional traces. Noticeable is the fact that adding information about the called functions, which is represented by the case study of the steering system, did not have a noticeable impact on how well the generated model reflects the actual timing behaviour of the system. The length and, thus, the amount of information contained within a trace recording plays a crucial role for the qual-

ity of CoreTAna's results, too. If a long period of the system's execution is recorded and used as input reflects the actual timing behaviour roughly five times more accurate. Regarding scalability, it takes CoreTAna to reversely engineer a model increases roughly linearly with the amount of events in a trace recording due to the fact that each event has to be processed once. Finally, some evaluation scenarios also highlighted CoreTAna's drawbacks. Because not only did missing information in the trace recording lead to large differences between the trace recordings of the actual system and those generated by simulating the synthesised model, but so did CoreTAna's lack in reversely engineering characteristics of the hardware platform such as the memory management unit. As a consequence, the resulting differences turned out to be too high to be useful for further processing. Our approach is aware of this drawback and can eliminate it if the user gives CoreTAna existing knowledge such as a model of the used hardware as input for processing the trace recording.

CoreTAna initiated our second main scientific contribution, DoTA, due to of the need to evaluate the quality of our reverse engineering approach. By choosing the Euclidean distance as the basis of our measure, DoTA is designed in an extensible way, which means that it can be customised arbitrarily by available real-time metrics. This is necessary if the trace recording lacks some information like the activation events, which are essential to determine activate-to-activate times. Another advantage of the Euclidean Distance is that the results of our measure show a predictable behaviour. The common architectural patterns in the real-time software domain and especially the feasible variations for each pattern that were elaborated for CoreTAna's evaluation confirmed this. Depending on how many entities and how many metrics are affected by a change, our measure yielded a corresponding value. For example, increasing the number of queued activation requests has an bigger impact on the patterns 'Client-Server without Reply' and 'State Machine' because they consist of only two tasks than the others that have at least five tasks. Our evaluation also showed that the measures of descriptive statistics, on which we applied the Euclidean distance, is a suitable method to determine even slight variations in the trace recordings such as the fluctuation of execution times. Also the quadratic characteristic of the used Euclidean distance helped us to highlight small changes. The assessment of CoreTAna's reverse engineering qualities is not the only application for which this metric can be used. During development, we recognised that DoTA provides a helpful way to assess changes between different versions of some real-time software, which allows one to comprehend their impact. Furthermore, we built DoTA into our internal trace re-

cording process, where our measure successfully checks the consistency of changes to trace configurations.

Summary

As a summary, the three research questions presented in Chapter 1.2 can be answered by the five formal contributions of this thesis:

- **Question Q1:** *To what extent is it possible to automatically synthesise an AUTOSAR-compliant model of a real-time system that covers the system's temporal behaviour based on event trace recordings?*
  Answer: This is shown by the algorithms defined in this work and implemented by CoreTAna (Contribution C1).
- **Question Q2:** *Can a synthesised model be validated with regards to the extent in which its representation reflects the temporal behaviour of the corresponding actual system?*
  Answer: On the one hand, it is possible by applying our defined approach that employs a model-based timing simulation to verify whether the simulation traces of the reversely engineered model and the hardware traces of the system under investigation show similar temporal behaviour (Contribution C2). On the other hand, it can be done by using our DoTA measure that expresses the accordance of two sample event trace recordings with respect to their represented temporal behaviour (Contribution C3).
- **Question Q3:** *To what extent is an approach for the automatic model synthesis of a real-time system from event trace recordings applicable to industrial projects in the automotive domain?*
  Answer: CoreTAna is an extensible realisation of the developed algorithms to automatically synthesise a probabilistic system model from event trace recordings that fits well within the AUTOSAR development process (Contribution C4). The presented case studies demonstrated not only the correctness of the developed algorithms but also the applicability and usefulness of the implementation for actual industrial projects (Contribution 5).

Although it has been proven that the proposed reverse engineering approach is ready to generate a model that is accurate enough to serve as documentation of a system's timing behaviour or to simulate and optimise the timing of a real-time system, there are further considerations possible, which are discussed next.

Future Work

We mentioned the drawback of our solution, namely that characteristics that are not supported by CoreTAna's reverse engineering algorithms result in high differences between the trace recordings of the actual system and those generated by simulating the synthesised model. Hence, applying such trace recordings to CoreTAna, e.g., one from a system whose communication is limited by a cross bar leads to poor models. The reason for this lies in the foundation of our approach. To give the users additional confidence in the results of CoreTAna, we decided to adopt a heuristic approach for our algorithms instead of using statistical techniques. This means that the AUTOSAR specification was converted to an algorithm for every element in the model that is reversely engineered by CoreTAna. This is possible because the standard not only consists of a well-defined meta-model but also of detailed documents that describe the semantics and constraints for the individual model elements. However, AUTOSAR has reached such a large scale since the release of version 3.0 in 2007 in consequence of the multitude of development steps that have to be covered and of the backward compatibility that has to be ensured and continues to grow further with every release that it would take an immense effort to cover the entire standard. For this reason, we propose to combine our reverse engineering algorithms with a meta-heuristic optimization algorithm [85] in the future. First, a model should be synthesised based on the existing knowledge from the AUTOSAR specifications, which conveys the essential confidence in the model. After that, an optimisation is started that tries to find a model whose simulation trace is closer to the actual timing behaviour of the system, by modifying individual characteristics of the model such as the task priorities. Such a search strategy allows one to further improve the results from a heuristic without loosing confidence in them.

Another aspect that has to be considered in more detail in the future is the reverse engineering of the hardware system on which the software is executed. The case study of the FMTV challenge in Chapter 8.3.1 highlights this fact, in which the accordance of the generated model to the actual timing behaviour of the system improved from 68 to 98.4 % by simply adding information on the hardware platform. Especially, the design of the hardware, including the location of memories and their connection to the processing cores, has a crucial impact on the timing behaviour of the system. Reversely engineering these hardware characteristics poses a major challenge because of the missing details on the internals of the processing unit in the trace recording. Thus, all aspects of the hardware platform have to be inferred from

existing information such as variations in the execution times. However, this also implies that all real-time metrics have to be adapted such that the overhead produced by the hardware is not included any more.

With the current progress in artificial intelligence and machine learning, another possible future research topic is the definition of an approach that does not completely rely on heuristics but that rather deduces information. This could, for example, enable the synthesis of model parts such as data dependencies for which usually a trace recording at system level is required, from a trace at process level. But there are also some model parts of the AUTOSAR model, e.g., design patterns such as a client-server communication or software components that are currently not considered because they cannot be identified clearly in a heuristic manner and, thus, would require the use of learning algorithms or statistical models.

Bibliography

[1] D. Kum, G.-M. Park, S. Lee and W. Jung, "AUTOSAR Migration from Existing Automotive Software", in *Intl. Conf. on Control, Automation and Systems*, IEEE, 2008, pp. 558–562.

[2] S. Anssi Saoussen and Tucci-Piergiovanni, S. Kuntz and F. Gerard Sebastien and Terrier, "Enabling Scheduling Analysis for AUTOSAR Systems", in *Intl. Symp. on Object/Component/Service-oriented Real-time Distributed Computing*, IEEE, 2011, pp. 152–159.

[3] A. Sailer, S. Schmidhuber, M. Deubzer, M. Alfranseder, M. Mucha and J. Mottok, "Optimizing the Task Allocation Step for Multi-Core Processors within AUTOSAR", in *Intl. Conf. on Applied Electronics*, IEEE, 2013, pp. 247–252.

[4] J. Kraft, "Enabling Timing Analysis of Complex Embedded Software Systems", Dissertation, Mälardalen University, 2010.

[5] M. Krogmann Klaus and Kuperberg and R. Reussner, "Using Genetic Search for Reverse Engineering of Parametric Behavior Models for Performance Prediction", *IEEE Trans. Softw. Eng.*, vol. 36, pp. 865–877, 6 2010.

[6] L. Thiele, S. Chakraborty and M. Naedele, "Real-Time Calculus for Scheduling Hard Real-Time Systems", in *Intl. Symposium on Circuits and Systems*, vol. 4, Geneva, CH: IEEE, May 2000, pp. 101–104.

[7] H. M. Kienle, J. Kraft and H. A. Müller, "Software Reverse Engineering in the Domain of Complex Embedded Systems", in *Re-*

verse Engineering - Recent Advances and Applications, A. C. Telea, Ed., InTech, Mar. 2012, ISBN: 978-953-51-0158-1.

[8] K. Krogmann, "Reconstruction of Software Component Architectures and Behaviour Models using Static and Dynamic Analysis", Dissertation, Karlsruher Institut für Technologie, 2012.

[9] J. Huselius, "Reverse Engineering of Legacy Real-Time Systems: An Automated Approach Based on Execution-Time Recording", Dissertation, Mälardalen University, 2007.

[10] P. Altenbernd, A. Ermedahl, B. Lisper and J. Gustafsson, "Automatic Generation of Timing Models for Timing Analysis of High-Level Code", in *Intl. Conf. on Real-Time Networks and Systems*, IRCCyN, 2011.

[11] F. Ciccozzi, "From Models to Code and Back: A Round-trip Approach for Model-driven Engineering of Embedded Systems", Dissertation, Mälardalen University, 2014.

[12] J. Andersson, J. Huselius, C. Nörström and A. Wall, "Extracting Simulation Models from Complex Embedded Real-Time Systems", in *Intl. Conf. on Software Engineering Advances*, IEEE, 2006.

[13] A. Sailer, M. Deubzer, G. Lüttgen and J. Mottok, "Comparing Trace Recordings of Automotive Real-time Software", in *Intl. Conf. on Real-Time Networks and Systems*, ACM, 2017.

[14] A. Sailer, S. Schmidhuber, M. Hempe, M. Deubzer and J. Mottok, "Distributed Multi-Core Development in the Automotive Domain – A Practical Comparison of ASAM MDX vs. AUTOSAR vs. AMALTHEA", in *First Multi-Core Safe and Software-intensive Systems Improvement Community Workshop*, VDE, 2016.

[15] A. Sailer, M. Deubzer, G. Lüttgen and J. Mottok, "CoreTAna: A Trace Analyzer for Reverse Engineering Real-Time Software", in *Intl. Conf. on Software Analysis, Evolution, and Reengineering*, IEEE, 2016, pp. 17–28.

[16] A. Sailer, "Towards an Automated Reverse Engineering of Design Models from Trace Recordings", in *Jahrestagung der Gesellschaft für Informatik*, GI, 2014, pp. 2233–2245.

[17] A. Sailer, S. Schmidhuber, M. Deubzer and J. Mottok, "AMALTHEA – Plattform für kontinuierliche, modellbasierte Entwicklung", in *Embedded Software Engineering Kongress*, ELEKTRONIKPRAXIS Vogel Business Media GmbH & Co. KG und MicroConsult Microelectronics Consulting & Training GmbH, 2013, pp. 538–544.

[18] P. Harrer, "Development of an Algorithm for Comparing Traces", Master's Thesis, Hochschule Nordhausen, 2016.

[19] F. Martin, "Transformation of Hardware Traces to System Traces for Embedded Multi-Core Real-Time Systems", Master's Thesis, Ostbayerische Technische Hochschule (OTH) Regensburg, 2015.

[20] X. Tang, "Trace-based Timing Verification of Real-Time Systems", Master's Thesis, University of Shanghai for Science and Technology, 2014.

[21] M. Alfranseder, M. Mucha, S. Schmidhuber, A. Sailer, M. Niemetz and J. Mottok, "A Modified Synchronization Model for Dead-lock Free Concurrent Execution of Strongly Interacting Task Sets in Embedded Systems", in *Intl. Conf. on Applied Electronics*, IEEE, 2013, pp. 13–18.

[22] J. Mottok, M. Alfranseder, S. Schmidhuber, M. Mucha and A. Sailer, "How to Improve the Reactiveness and Efficiency of Embedded Multi-core Systems by Use of Probabilistic Simulation and Optimization Techniques", in *NATO Advanced Research Workshop on Improving Disaster Resilience and Mitigation – IT Means and Tools*, Springer, 2013, ch. 16, pp. 253–268.

[23] A. W. Biermann, "On the Inference of Turing Machines from Sample Computations", *Artificial Intelligence*, vol. 3, pp. 181–198, 1972.

[24] B. Cornelissen, A. Zaidman, A. van Deursen, L. Moonen and R. Koschke, "A Systematic Survey of Program Comprehension through Dynamic Analysis", *Trans. Softw. Eng.*, vol. 35, no. 5, pp. 684–702, 2009.

[25] J. Huselius and J. Andersson, "Model Synthesis for Real-Time Systems", in *European Conf. of Software Maintenance and Reengineering*, Manchester, UK: IEEE, 2005, pp. 52–60.

[26] J. Huselius, H. Hansson and S. Punnekkat, "Presenting: An Automated Process for Model Synthesis", Mälardalen University, Västeras, SE, MRTC Report, Oct. 2005.

[27] J. Huselius, J. Andersson, H. Hansson and S. Punnekkat, "Automatic Generation and Validation of Models of Legacy Software", in *Intl. Conf. on Embedded and Real-Time Computing Systems and Applications*, Sydney, AU: IEEE, 2006, pp. 342–349.

[28] J. Andersson, "Modeling the Temporal Behavior of Complex Embedded Systems: A Reverse Engineering Approach", Licentiate Thesis, Mälardalen University, 2005.

[29] J. Sifakis, S. Tripakis and S. Yovine, "Building Models of Real-Time Systems from Application Software", *Proc. IEEE*, vol. 91, no. 1, pp. 100–111, 2003.

[30] S. Chakraborty, S. Künzli and L. Thiele, "A General Framework for Analysing System Properties in Platform-Based Embedded System Designs", in *Design, Automation and Test in Europe*, Munich, DE: IEEE, 2003.

[31] E. Wandeler, "Modular Performance Analysis and Interface-Based Design for embedded Real-Time Systems", Dissertation, Swiss Federal Institute of Technology (ETH), 2006.

[32] F. Ciccozzi, A. Cicchetti and M. Sjödin, "Towards a Round-Trip Support for Model-Driven Engineering of Embedded Systems", in *EUROMICRO Copnf. Series on Software Engineering and Advanced Applications*, Oulu, FI: IEEE, 2011.

[33] F. Ciccozzi, M. Saadatmand, A. Cicchetti and M. Sjödin, "An Automated Round-Trip Support Towards Deployment Assessment in Component-based Embedded Systems", in *Intl. ACM Sigsoft Symposium on Component-Based Software Engineering*, Vancouver, CA: ACM, 2013.

[34] A. Terrasa and G. Bernat, "Extracting Temporal Properties from Real-Time Systems by Automatic Tracing Analysis", in *Intl. Conf. on Embedded and Real-Time Computing Systems and Applications*, IEEE, 2004, pp. 466–485.

[35] M. Auguston, "Building Program Behavior Models", in *Engineering Automation for Reliable Software - Interrim Progress Report*, Naval Postgraduate School Monterey, Ed., Research Triangle Park, NC: US. Army Research Office, 2000, pp. 35–55.

[36] D. Murphy Gail C. and Notkin, "Reengineering with Reflexion Models: A Case Study", *Computer*, vol. 30, pp. 29–37, 8 1997.

[37] D. Murphy Gail C. and Notkin and K. J. Sullivan, "Software Reflexion Models: Bridging the Gap between Design and Implementation", *Transactions on Software Engineering*, vol. 27, pp. 364–380, 4 2001.

[38] A. van Hoorn, J. Waller and W. Hasselbring, "Kieker: A Framework for Application Performance Monitoring and Dynamic Software Analysis", in *Intl. Conf. on Performance Engineering*, Boston, MA: ACM, 2012.

[39] J. E. Cook and A. L. Wolf, "Automating Process Discovery through Event-Data Analysis", in *Intl. Conf. on Software Engineering*, IEEE, 1995, pp. 73–73.

[40] J. E. Cook and A. L. Wolf, "Event-Based Detection of Concurrency", *SIGSOFT Software Engineering Notes*, vol. 23, no. 6, pp. 35–45, 1998.

[41] W. van der Aalst, T. Weijters and L. Maruster, "Workflow Mining: Discovering Process Models from Event Logs", *Trans. on Knowledge and Data Engineering*, vol. 16, pp. 1128–1142, 9 2004.

[42] L. Wen, J. Wang, W. M. van der Aalst, B. Huang and J. Sun, "A novel Approach for Process Mining Based on Event Types", *J. of Intelligent Information Systems*, vol. 32, no. 2, pp. 163–190, 2009.

[43] J. Kraft, A. Wall and H. Kienle, "Trace recording for embedded systems: Lessons learned from five industrial projects", in *Intl. Conf. on Runtime Verification*, ser. Lecture Notes in Computer Science, St. Julians, MT: Springer, Nov. 2010, pp. 315–329.

[44] S. Künzli and L. Thiele, "Generating Event Traces Based on Arrival Curves", in *Conf. on Measurement, Modelling and Evaluation of Computer and Communication Systems*, Nuremberg: VDE, Mar. 2006, pp. 1–18.

[45] "Der neue Maybach", *ATZ/MTZ extra*, p. 125, 2002.

[46] I. Foster, *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*, 1st ed. Addison-Wesley, 1995, ISBN: 0-201-57594-9.

[47] B. Blaise. (2018). Introduction to Parallel Computing. Last Accessed: February 18, 2018, [Online]. Available: `https://computing.llnl.gov/tutorials/parallel_comp/`.

[48] L. Michel, T. Flaemig, D. Claraz and R. Mader, "Shared SW development in multi-core automotive context", in *European Congress on Embedded Real Time Software and Systems (ERTS 2016)*, 2016.

[49] M. Deubzer, "Robust Scheduling of Real-Time Applications on Efficient Embedded Multicore Systems", Dissertation, Technische Universität München, 2011.

[50] A. Burns and A. Wellings, *Real-Time Systems and Programming Languages*, 3rd ed. Pearson Education Limited, 2001.

[51] "Encyclopedia of Mathematics", in, M. Hazewinkel, Ed., 1st ed. Springer Science+Business Media B.V., 1994, ch. Weibull distribution, ISBN: 978-1-55608-010-4.

[52] "Encyclopedia of Mathematics", in, M. Hazewinkel, Ed., 1st ed. Springer Science+Business Media B.V., 1994, ch. Beta-distribution, ISBN: 978-1-55608-010-4.

[53] ASAM e.V., *ASAM AE MDX – Metadata Exchange Format for Software Module Sharing*, Version 1.3.0, Jun. 2015.

[54] AUTOSAR, *Software Component Template*, V 4.2.2, Jul. 2015.

[55] AUTOSAR, *Specification of ECU Resource Template*, V 4.2.2, Jul. 2015.

[56] AUTOSAR, *Specification of Timing Extensions*, V 4.2.2, Jul. 2015.

[57] AMALTHEA, *AMALTHEA Documentation*, V1.1.1, Nov. 2015.

[58] Eclipse Foundation. (2018). Application Platform Project for MultiCore (APP4MC). Last Accessed: February 26, 2018, [Online]. Available: `https://www.eclipse.org/app4mc/`.

[59] Timing-Architects Embedded Systems GmbH. (2016). BTF-Specification (Version 2.1.5). Last Accessed: July 13, 2017, [Online]. Available: `https://www.eclipse.org/app4mc/docu/standards/TA_BTF_Specification_2.1.5.pdf`.

[60] D. Ferrari, *Computer systems performance evaluation*, 1st ed. Prentice Hall, 1978, ISBN: 0-13-165126-9.

[61] S. Rafiq and A. Schmidt, "Systematic Modeling of Workflows in Trace-Based Software Debugging and Optimization", in *Intl. Conf. on Software Engineering Advances*, Venice, IT: IARIA, Oct. 2013, pp. 241–248.

[62] J. Trümper, S. Voigt and J. Döllner, "Maintenance of Embedded Systems: Supporting Program Comprehension Using Dynamic Analysis", in *Intl. Workshop on Software Engineering for Embedded Systems*, Zurich, CH: IEEE, Jun. 2012, pp. 58–64.

[63] International Organization for Standardization, "Road vehicles - Functional safety", no. ISO 26262, Nov. 2011.

[64] Stahleder, Elmar. (2005). Debugger mit Rückspiegel - Trace Techniken im Überblick. Last Accessed: November 12, 2017, [Online]. Available: http://www.lauterbach.com/publications/trace_methoden_d.pdf.

[65] AUTOSAR, *Specification of Operating System*, V 4.2.2, Jul. 2015.

[66] OSEK/VDX, *Operating System Specification*, Version 2.2.3, Jun. 2005.

[67] AUTOSAR, *Specification of RTE*, V 4.2.2, Jul. 2015.

[68] A. Hamann, D. Ziegenbein, S. Kramer and M. Lukasiewycz. (2016). 7th Intl. Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems – Verification Challenge. Last Accessed: March 24, 2017, [Online]. Available: https://waters2016.inria.fr/challenge/.

[69] G. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, 3rd ed. Springer, 2011.

[70] Timing-Architects Embedded Systems GmbH. (2016). TA Toolsuite Version 16.03.0. TA Academic & Research License Program. Last Accessed: March 24, 2017, [Online]. Available: http://www.timing-architects.com.

[71] B. Rüger, *Test- und Schätztheorie – Band II: Statistische Tests*, 1st ed. Oldenburg, 2002, ISBN: 3-486-25130-9.

[72] AUTOSAR, *Specification of BSW Module Description Template*, V 4.2.2, Jul. 2015.

[73] Y. Lu, T. Nolte, I. Bate, J. Kraft and C. Norström, "Assessment of Trace-Differences in Timing Analysis for Complex Real-Time Embedded Systems", in *Intl. Symposium on Industrial Embedded Systems*, Västeras, SE: IEEE, Jun. 2011, pp. 284–293.

[74] "Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice", in, J. Banks, Ed., 1st ed. Wiley, 1998, ch. 10) Verification, Validation and Testing, ISBN: 978-0-471-13403-9.

[75] R. G. Sargent, "Verification and Validation of Simulation Models", in *Winter Simulation Conf.*, Washington, D.C.: IEEE, Dec. 2007, pp. 124–137.

[76] A. M. Law, "How to Build Valid and Credible Simulation Models", in *Winter Simulation Conf.*, Austin, TX: IEEE, Dec. 2009, pp. 24–33.

[77] J. Huselius, J. Kraft, H. Hansson and S. Punnekkat, "Evaluating the Quality of Models Extracted from Embedded Real-Time Software", in *Intl. Conf. and Workshops on the Engineering of Computer-Based Systems*, Tucson, AZ: IEEE, 2007, pp. 577–585.

[78] A. V. Miranskyy, M. Davison, M. R. Reesor and S. S. Murtaza, "Using Entropy Measures for Comparison of Software Traces", *Information Sciences*, vol. 203, pp. 59–72, 2012.

[79] F. Nemati, J. Kraft and C. Norström, "Validation of Temporal Simulation Models of Complex Real-Time Systems", in *Intl. Computers, Software and Applications Conf.*, Turku, FI: IEEE, Jul. 2008, pp. 1335–1340.

[80] A. Hamou-Lhadj, "Measuring the Complexity of Traces Using Shannon Entropy", in *Intl. Conf. on Information Technology: New Generations*, IEEE, 2008, pp. 489–494.

[81] "Encyclopedia of Mathematics", in, M. Hazewinkel, Ed., 1st ed. Springer Science+Business Media B.V., 1994, ch. Pearson product-moment correlation coefficient, ISBN: 978-1-55608-010-4.

[82] AUTOSAR, *Methodology*, V 4.2.2, Jul. 2015.

[83] B. Steppan, *Eclipse Rich Clients und Plug-ins: Modulare Desktop-Anwendungen mit Java entwickeln*, 1st ed. Hanser, 2015, ISBN: 978-3-446-43172-0.

[84] P. Hofstedt and A. Wolf, *Einführung in die Constraint-Programmierung. Grundlagen, Methoden, Sprachen, Anwendungen*, 1st ed. Springer, 2007, ISBN: 978-3-540-68194-6.

[85] "Artificial Intelligence: A Modern Approach", in, S. Russel and P. Norvig, Eds., 3rd ed. Pearson, 2016, ch. Local Search Algorithms and Optimization Problems, ISBN: 978-1292153964.

Acronyms

χ^2 Test Chi-squared Test, *see also Glossary:* Chi-squared Test.

APP4MC Application Platform Project for Multi-Core, *see also Glossary:* APP4MC.

ASAM Association for Standardization of Automation and Measuring Systems.

ATDB AMALTHEA trace database.

AUTOSAR AUTomotive Open System ARchitecture, *see also Glossary:* AUTOSAR.

BTF Best Trace Format, *see also Glossary:* BTF.

CoreTAna Core Trace Analyser, *see also Glossary:* CoreTAna.

CSP Constraint Satisfaction Problem, *see also Glossary:* CSP.

CSV comma-separated values.

DoTA Distance of Timed Actions, *see also Glossary:* DoTA.

ECU Electronic Control Unit.

ET Execution Time, *see also Glossary:* ET.

GET Gross Execution Time.

ISR Interrupt Service Routine.

K-S Test Kolmogorov-Smirnov Test, *see also Glossary:* Kolmogorov-Smirnov Test.

LCM least common multiple.

MDX Model Data Exchange Format, *see also Glossary:* MDX.

NET Net Execution Time.

OEM Original Equipment Manufacturer, *see also Glossary:* OEM.

OS Operating System.

PIC Population, Imperfectness, or Complexity.

RT Response Time, *see also Glossary:* RT.

RTE Runtime Environment.

SoD Sum of Divergence, *see also Glossary:* Sum of Divergence.

TA Timing-Architects.

WCET Worst Case Execution Time.

Glossary

AMALTHEA is a data model originally defined by the ITEA research projects AMALTHEA and AMALTHEA4public. It is maintained within the Eclipse project APP4MC and focuses on design, implementation and optimization of software for multi- and many-core real-time systems.

APP4MC is an open source project hosted by the Eclipse Foundation. Goal of the project is to develop a platform for engineering embedded multi- and many-core software systems in order to support the interoperability and extensibility and to unify data exchange in cross-organizational projects.

AUTOSAR is a cooperation of automotive companies that aims to improve complexity management of integrated E/E architectures though model-based development. They standardise a software architecture and methodology to increase reuse and exchangeability of software modules between OEMs and suppliers.

BTF is a comma-separated values (CSV)-based format for representation of event-traces in ASCII. It is designed for analysing the behaviour of a system in an efficient and chronologically correct manner in order to apply timing, performance, or reliability evaluations. The standard was initially defined by Continental Automotive GmbH and is now publicly available on the Eclipse APP4MC website.

Chi-squared Test is also written as χ^2 Test and is a statistical goodness-of-fit test that evaluates how much an observed frequency distribution differs from a theoretical distribution.

CoreTAna is a tool that derives an AUTOSAR-compliant model of a real-time system by conducting dynamic analysis using trace recordings.

CSP is a mathematical definition of a solution space that is limited by a number of constraints.

DoTA is a measure based on the Euclidean distance that determines the difference between trace recordings based on real-time metrics..

ET defines the time interval between the start and the termination of a process. Its two manifestations gross and net execution time distinguish whether a process was actually executing on a core or not.

Kolmogorov-Smirnov Test is also known as K-S Test and is a statistical goodness-of-fit test that evaluates whether a random variable follows a specific statistical distribution.

MDX is a model for data management and documentation standardised by the Association for Standardization of Automation and Measuring Systems (ASAM).

OEM refers in the automotive supply chain to a company that is manufacturing cars.

PATRICIA Trie PATRICIA (Practical Algorithm to Retrieve Information Coded in Alphanumeric) trie, which is also know as radix trie or compact prefix tree, is an optimised data structure for retrieving strings with a common prefix. .

RT defines the time interval between the activation and the termination of a process.

Runnable is also known as Runnable Entity and is used interchangeably to software function in automotive terminology. It represents a sequence of instructions which can smallest unit of a software component that can be schedules independently.

Sum of Divergence is a measure defined by J. Huselius that determines the difference between two series of sampled response times.

TA Tool Suite is collection of tools developed by Timing-Architects Embedded Systems GmbH for designing, developing and verifying embedded multi- and many-core systems or trace recording.

Tier-1 refers in the automotive supply chain to a company that supplies parts or systems directly to OEMs.

Trace or trace recording is a sequence of events that have been detected and stored during runtime of a system for later off-line analysis.

List of Figures

List of Tables

Listings

List of Algorithms

Part IV.

Annex

A

Appendix

A.1. Architectural System Patterns

A.1.1. Purely Periodic without Communication

A.1.1.1. Variation 1

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
    </tasks>
    <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
    </tasks>
    <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
```

```
32        <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
34        </graphEntries>
        </callGraph>
36      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="2" />
38      </customProperties>
        <customProperties key="osekTaskGroup">
40        <value xsi:type="am:StringObject" value="2" />
        </customProperties>
42    </tasks>
      <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
44      <callGraph>
          <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
46          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
48        </graphEntries>
        </callGraph>
50      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
52      </customProperties>
        <customProperties key="osekTaskGroup">
54        <value xsi:type="am:StringObject" value="1" />
        </customProperties>
56    </tasks>
      <runnables name="Runnable_4" callback="false" service="false">
58      <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
60          <value xsi:type="am:NeedDeviation">
              <deviation>
62              <lowerBound xsi:type="am:LongObject" value="8970000" />
                <upperBound xsi:type="am:LongObject" value="9000000" />
64              <distribution xsi:type="am:UniformDistribution" />
              </deviation>
66          </value>
          </default>
68      </runnableItems>
      </runnables>
70    <runnables name="Runnable_5" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
72        <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
74            <deviation>
                <lowerBound xsi:type="am:LongObject" value="17970000" />
76              <upperBound xsi:type="am:LongObject" value="18000000" />
                <distribution xsi:type="am:UniformDistribution" />
78            </deviation>
            </value>
80        </default>
        </runnableItems>
82    </runnables>
      <runnables name="Runnable_6" callback="false" service="false">
84      <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
86          <value xsi:type="am:NeedDeviation">
              <deviation>
88              <lowerBound xsi:type="am:LongObject" value="23970000" />
```

```
            <upperBound xsi:type="am:LongObject" value="24000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_7_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="35977500" />
            <upperBound xsi:type="am:LongObject" value="36000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_7_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="11992500" />
            <upperBound xsi:type="am:LongObject" value="12000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
</swModel>
<hwModel>
  <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
      IPC_1.0?type=HwFeature" puType="CPU"/>
  <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
  </definitions>
  <featureCategories name="Instructions" featureType="performance">
    <features name="IPC_1.0" value="1.0" />
  </featureCategories>
  <structures name="System" structureType="System">
    <structures name="Ecu_1" structureType="ECU">
      <structures name="Processor_1" structureType="Microcontroller">
        <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
        </modules>
        <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
          <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
        </modules>
      </structures>
    </structures>
  </structures>
  <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
    <defaultValue value="600.0" unit="MHz"/>
  </domains>
```

```
144   </hwModel>
      <osModel>
146     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
148         <schedulingAlgorithm xsi:type="am:OSEK" />
          </taskSchedulers>
150       <osDataConsistency mode="noProtection" />
        </operatingSystems>
152   </osModel>
      <stimuliModel>
154     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
          <offset value="0" unit="ms" />
156       <recurrence value="180" unit="ms" />
        </stimuli>
158     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
          <offset value="0" unit="ms" />
160       <recurrence value="200" unit="ms" />
        </stimuli>
162     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
          <offset value="0" unit="ms" />
164       <recurrence value="300" unit="ms" />
        </stimuli>
166     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
          <offset value="0" unit="ms" />
168       <recurrence value="1000" unit="ms" />
        </stimuli>
170   </stimuliModel>
      <constraintsModel />
172   <eventModel>
        <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
174     <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
        <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
176     <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4" entity="Runnable_4?type=Runnable"
            />
178     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
            />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6" entity="Runnable_6?type=Runnable"
            />
180     <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
            Runnable" />
182     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
            PeriodicStimulus" />
184     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="" entity="
            Stimulus_Task_6?type=PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
            PeriodicStimulus" />
186   </eventModel>
      <mappingModel addressMappingType="offset">
188     <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
190     <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
```

```
192     <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
    </mappingModel>
194   <componentsModel />
  </am:Amalthea>
```

Listing A.1: Variation 1 of Purely Periodic without Communication.

A.1.1.2. Variation 2

```
  <?xml version="1.0" encoding="UTF-8"?>
2 <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
      " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
    <swModel>
4     <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
        <callGraph>
6         <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
8         </graphEntries>
        </callGraph>
10      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="5" />
12      </customProperties>
        <customProperties key="osekTaskGroup">
14        <value xsi:type="am:StringObject" value="5" />
        </customProperties>
16    </tasks>
      <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
18      <callGraph>
          <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
20          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
          </graphEntries>
22      </callGraph>
        <customProperties key="priority">
24        <value xsi:type="am:StringObject" value="4" />
        </customProperties>
26      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="4" />
28      </customProperties>
      </tasks>
30    <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
        <callGraph>
32        <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
34        </graphEntries>
        </callGraph>
36      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="3" />
38      </customProperties>
        <customProperties key="osekTaskGroup">
40        <value xsi:type="am:StringObject" value="3" />
        </customProperties>
42    </tasks>
```

```
    <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
    <runnables name="Runnable_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11970000" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="8970000" />
              <upperBound xsi:type="am:LongObject" value="9000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_5" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
```

```
          <value xsi:type="am:NeedDeviation">
100         <deviation>
              <lowerBound xsi:type="am:LongObject" value="17970000" />
102           <upperBound xsi:type="am:LongObject" value="18000000" />
              <distribution xsi:type="am:UniformDistribution" />
104         </deviation>
          </value>
106     </default>
      </runnableItems>
108   </runnables>
    <runnables name="Runnable_6" callback="false" service="false">
110   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
112       <value xsi:type="am:NeedDeviation">
            <deviation>
114           <lowerBound xsi:type="am:LongObject" value="23970000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
116           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
118       </value>
        </default>
120   </runnableItems>
    </runnables>
122 <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
124     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
126         <deviation>
              <lowerBound xsi:type="am:LongObject" value="35977500" />
128           <upperBound xsi:type="am:LongObject" value="36000000" />
              <distribution xsi:type="am:UniformDistribution" />
130         </deviation>
          </value>
132     </default>
      </runnableItems>
134 </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
136   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
138       <value xsi:type="am:NeedDeviation">
            <deviation>
140           <lowerBound xsi:type="am:LongObject" value="11992500" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
142           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
144       </value>
        </default>
146   </runnableItems>
    </runnables>
148 </swModel>
    <hwModel>
150   <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
          IPC_1.0?type=HwFeature" puType="CPU"/>
      <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
152   </definitions>
      <featureCategories name="Instructions" featureType="performance">
154     <features name="IPC_1.0" value="1.0" />
      </featureCategories>
```

```
156     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
158         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
160           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
162             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
164         </structures>
          </structures>
166     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
168       <defaultValue value="600.0" unit="MHz"/>
        </domains>
170   </hwModel>
      <osModel>
172     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
174         <schedulingAlgorithm xsi:type="am:OSEK" />
          </taskSchedulers>
176       <osDataConsistency mode="noProtection" />
        </operatingSystems>
178   </osModel>
      <stimuliModel>
180     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
          <offset value="0" unit="ms" />
182       <recurrence value="160" unit="ms" />
        </stimuli>
184     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
          <offset value="0" unit="ms" />
186       <recurrence value="180" unit="ms" />
        </stimuli>
188     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
          <offset value="0" unit="ms" />
190       <recurrence value="200" unit="ms" />
        </stimuli>
192     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
          <offset value="0" unit="ms" />
194       <recurrence value="300" unit="ms" />
        </stimuli>
196     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
          <offset value="0" unit="ms" />
198       <recurrence value="1000" unit="ms" />
        </stimuli>
200   </stimuliModel>
      <constraintsModel />
202   <eventModel>
        <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
204     <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
        <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
206     <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
        <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
208     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
            />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4" entity="Runnable_4?type=Runnable"
            />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6" entity="Runnable_6?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
        Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="" entity="
        Stimulus_Task_6?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.2: Variation 2 of Purely Periodic without Communication.

A.1.1.3. Variation 3

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="7" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="7" />
      </customProperties>
    </tasks>
    <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
```

```
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
```

```
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
74      </graphEntries>
      </callGraph>
76    <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
78    </customProperties>
      <customProperties key="osekTaskGroup">
80      <value xsi:type="am:StringObject" value="1" />
      </customProperties>
82  </tasks>
    <runnables name="Runnable_1" callback="false" service="false">
84    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
86        <value xsi:type="am:NeedDeviation">
            <deviation>
88            <lowerBound xsi:type="am:LongObject" value="5970000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
90            <distribution xsi:type="am:UniformDistribution" />
            </deviation>
92        </value>
        </default>
94    </runnableItems>
  </runnables>
96  <runnables name="Runnable_3" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
98      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
100       <deviation>
            <lowerBound xsi:type="am:LongObject" value="11970000" />
102         <upperBound xsi:type="am:LongObject" value="12000000" />
            <distribution xsi:type="am:UniformDistribution" />
104       </deviation>
        </value>
106     </default>
    </runnableItems>
108 </runnables>
    <runnables name="Runnable_4" callback="false" service="false">
110   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
112       <value xsi:type="am:NeedDeviation">
            <deviation>
114           <lowerBound xsi:type="am:LongObject" value="8970000" />
              <upperBound xsi:type="am:LongObject" value="9000000" />
116           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
118       </value>
        </default>
120   </runnableItems>
    </runnables>
122 <runnables name="Runnable_5" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
124     <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
126       <deviation>
            <lowerBound xsi:type="am:LongObject" value="17970000" />
128         <upperBound xsi:type="am:LongObject" value="18000000" />
            <distribution xsi:type="am:UniformDistribution" />
130       </deviation>
```

```
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23970000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="35977500" />
              <upperBound xsi:type="am:LongObject" value="36000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11992500" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
```

```
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="80" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
      <recurrence value="160" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
      <offset value="0" unit="ms" />
      <recurrence value="180" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="200" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
      <offset value="0" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
      <recurrence value="1000" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3">
      <entity xsi:type="am:Task" href="amlt:/#Task_3?type=Task" />
    </events>
    <events xsi:type="am:ProcessEvent" name="Event_Task_4">
      <entity xsi:type="am:Task" href="amlt:/#Task_4?type=Task" />
    </events>
    <events xsi:type="am:ProcessEvent" name="Event_Task_5">
      <entity xsi:type="am:Task" href="amlt:/#Task_5?type=Task" />
    </events>
    <events xsi:type="am:ProcessEvent" name="Event_Task_6">
```

```
244       <entity xsi:type="am:Task" href="amlt:/#Task_6?type=Task" />
        </events>
246     <events xsi:type="am:ProcessEvent" name="Event_Task_7">
          <entity xsi:type="am:Task" href="amlt:/#Task_7?type=Task" />
248     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
            />
250     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3">
          <entity href="amlt:/#Runnable_3?type=Runnable" />
252     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4">
254       <entity href="amlt:/#Runnable_4?type=Runnable" />
        </events>
256     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5">
          <entity href="amlt:/#Runnable_5?type=Runnable" />
258     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6">
260       <entity href="amlt:/#Runnable_6?type=Runnable" />
        </events>
262     <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1">
          <entity href="amlt:/#Runnable_7_1?type=Runnable" />
264     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2">
266       <entity href="amlt:/#Runnable_7_2?type=Runnable" />
        </events>
268     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3">
270       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_3?type=PeriodicStimulus" /
              >
        </events>
272     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_4?type=PeriodicStimulus" /
              >
274     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5">
276       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_5?type=PeriodicStimulus" /
              >
        </events>
278     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_6?type=PeriodicStimulus" /
              >
280     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7">
282       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_7?type=PeriodicStimulus" /
              >
        </events>
284   </eventModel>
      <mappingModel addressMappingType="offset">
286     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
288     <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
290     <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
292     <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
      </mappingModel>
```

```
294   <componentsModel />
</am:Amalthea>
```

Listing A.3: Variation 3 of Purely Periodic without Communication.

A.1.1.4. Variation 4

```
<?xml version="1.0" encoding="UTF-8"?>
2 <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
      " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
    <swModel>
4     <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
        <callGraph>
6         <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
8         </graphEntries>
        </callGraph>
10      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="7" />
12      </customProperties>
        <customProperties key="osekTaskGroup">
14        <value xsi:type="am:StringObject" value="7" />
        </customProperties>
16    </tasks>
      <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
18      <callGraph>
          <graphEntries xsi:type="am:CallSequence" name="CallSequence_2_1">
20          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2?type=Runnable" />
          </graphEntries>
22      </callGraph>
        <customProperties key="priority">
24        <value xsi:type="am:StringObject" value="6" />
        </customProperties>
26      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="6" />
28      </customProperties>
      </tasks>
30    <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
        <callGraph>
32        <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
34        </graphEntries>
        </callGraph>
36      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="5" />
38      </customProperties>
        <customProperties key="osekTaskGroup">
40        <value xsi:type="am:StringObject" value="5" />
        </customProperties>
42    </tasks>
      <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
44      <callGraph>
```

```
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
```

```
100           <deviation>
                <lowerBound xsi:type="am:LongObject" value="5970000" />
102             <upperBound xsi:type="am:LongObject" value="6000000" />
                <distribution xsi:type="am:UniformDistribution" />
104           </deviation>
            </value>
106       </default>
        </runnableItems>
108   </runnables>
      <runnables name="Runnable_2" callback="false" service="false">
110     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
112         <value xsi:type="am:NeedDeviation">
              <deviation>
114             <lowerBound xsi:type="am:LongObject" value="17970000" />
                <upperBound xsi:type="am:LongObject" value="18000000" />
116             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
118         </value>
          </default>
120     </runnableItems>
      </runnables>
122   <runnables name="Runnable_3" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
124       <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
126           <deviation>
                <lowerBound xsi:type="am:LongObject" value="11970000" />
128             <upperBound xsi:type="am:LongObject" value="12000000" />
                <distribution xsi:type="am:UniformDistribution" />
130           </deviation>
            </value>
132       </default>
        </runnableItems>
134   </runnables>
      <runnables name="Runnable_4" callback="false" service="false">
136     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
138         <value xsi:type="am:NeedDeviation">
              <deviation>
140             <lowerBound xsi:type="am:LongObject" value="8970000" />
                <upperBound xsi:type="am:LongObject" value="9000000" />
142             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
144         </value>
          </default>
146     </runnableItems>
      </runnables>
148   <runnables name="Runnable_5" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
150       <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
152           <deviation>
                <lowerBound xsi:type="am:LongObject" value="17970000" />
154             <upperBound xsi:type="am:LongObject" value="18000000" />
                <distribution xsi:type="am:UniformDistribution" />
156           </deviation>
            </value>
```

```
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23970000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="35977500" />
              <upperBound xsi:type="am:LongObject" value="36000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11992500" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
```

```
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
214         <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
216     </structures>
      </structures>
218   </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
220   <defaultValue value="600.0" unit="MHz"/>
    </domains>
222 </hwModel>
  <osModel>
224   <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
226     <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
228   <osDataConsistency mode="noProtection" />
    </operatingSystems>
230 </osModel>
  <stimuliModel>
232   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
234   <recurrence value="80" unit="ms" />
    </stimuli>
236 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="0" unit="ms" />
238   <recurrence value="120" unit="ms" />
    </stimuli>
240 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
242   <recurrence value="160" unit="ms" />
    </stimuli>
244 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
      <offset value="0" unit="ms" />
246   <recurrence value="180" unit="ms" />
    </stimuli>
248 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
250   <recurrence value="200" unit="ms" />
    </stimuli>
252 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
      <offset value="0" unit="ms" />
254   <recurrence value="300" unit="ms" />
    </stimuli>
256 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
258   <recurrence value="1000" unit="ms" />
    </stimuli>
260 </stimuliModel>
  <constraintsModel />
262 <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1">
264   <entity xsi:type="am:Task" href="amlt:/#Task_1?type=Task" />
    </events>
266 <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3">
268   <entity xsi:type="am:Task" href="amlt:/#Task_3?type=Task" />
    </events>
```

```
270     <events xsi:type="am:ProcessEvent" name="Event_Task_4">
          <entity xsi:type="am:Task" href="amlt:/#Task_4?type=Task" />
272     </events>
        <events xsi:type="am:ProcessEvent" name="Event_Task_5">
274       <entity xsi:type="am:Task" href="amlt:/#Task_5?type=Task" />
        </events>
276     <events xsi:type="am:ProcessEvent" name="Event_Task_6">
          <entity xsi:type="am:Task" href="amlt:/#Task_6?type=Task" />
278     </events>
        <events xsi:type="am:ProcessEvent" name="Event_Task_7">
280       <entity xsi:type="am:Task" href="amlt:/#Task_7?type=Task" />
        </events>
282     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1">
          <entity href="amlt:/#Runnable_1?type=Runnable" />
284     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_2" entity="Runnable_2?type=Runnable"
            />
286     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3">
          <entity href="amlt:/#Runnable_3?type=Runnable" />
288     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4">
290       <entity href="amlt:/#Runnable_4?type=Runnable" />
        </events>
292     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5">
          <entity href="amlt:/#Runnable_5?type=Runnable" />
294     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6">
296       <entity href="amlt:/#Runnable_6?type=Runnable" />
        </events>
298     <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1">
          <entity href="amlt:/#Runnable_7_1?type=Runnable" />
300     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2">
302       <entity href="amlt:/#Runnable_7_2?type=Runnable" />
        </events>
304     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_1?type=PeriodicStimulus" /
              >
306     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
            PeriodicStimulus" />
308     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_3?type=PeriodicStimulus" /
              >
310     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4">
312       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_4?type=PeriodicStimulus" /
              >
        </events>
314     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_5?type=PeriodicStimulus" /
              >
316     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="">
318       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_6?type=PeriodicStimulus" /
              >
        </events>
320     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7">
```

```
      <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_7?type=PeriodicStimulus" />
    </events>
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.4: Variation 4 of Purely Periodic without Communication.

A.1.1.5. Variation 5

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="7" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="7" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_2_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="6" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="6" />
      </customProperties>
    </tasks>
    <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
```

```
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="
      non_preemptive" multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
```

```
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
    <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5970000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="17970000" />
              <upperBound xsi:type="am:LongObject" value="18000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11970000" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="8970000" />
              <upperBound xsi:type="am:LongObject" value="9000000" />
              <distribution xsi:type="am:UniformDistribution" />
```

```
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_5" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="17970000" />
              <upperBound xsi:type="am:LongObject" value="18000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23970000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="35977500" />
              <upperBound xsi:type="am:LongObject" value="36000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11992500" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
  </swModel>
```

```
<hwModel>
  <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
      IPC_1.0?type=HwFeature" puType="CPU"/>
  <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
  </definitions>
  <featureCategories name="Instructions" featureType="performance">
    <features name="IPC_1.0" value="1.0" />
  </featureCategories>
  <structures name="System" structureType="System">
    <structures name="Ecu_1" structureType="ECU">
      <structures name="Processor_1" structureType="Microcontroller">
        <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
        </modules>
        <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
          <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
        </modules>
      </structures>
    </structures>
  </structures>
  <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
    <defaultValue value="600.0" unit="MHz"/>
  </domains>
</hwModel>
<osModel>
  <operatingSystems name="Generic_OS">
    <taskSchedulers name="Scheduler_1">
      <schedulingAlgorithm xsi:type="am:OSEK" />
    </taskSchedulers>
    <osDataConsistency mode="noProtection" />
  </operatingSystems>
</osModel>
<stimuliModel>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
    <offset value="0" unit="ms" />
    <recurrence value="80" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
    <offset value="0" unit="ms" />
    <recurrence value="120" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
    <offset value="0" unit="ms" />
    <recurrence value="160" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
    <offset value="0" unit="ms" />
    <recurrence value="180" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
    <offset value="0" unit="ms" />
    <recurrence value="200" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
    <offset value="0" unit="ms" />
    <recurrence value="300" unit="ms" />
  </stimuli>
```

```
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
      <recurrence value="1000" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2" entity="Runnable_2?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4" entity="Runnable_4?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6" entity="Runnable_6?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
        Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="" entity="
        Stimulus_Task_6?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.5: Variation 5 of Purely Periodic without Communication.

A.1.1.6. Variation 6

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="7" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="7" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_2_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="6" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="6" />
      </customProperties>
    </tasks>
    <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="5" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="5" />
      </customProperties>
    </tasks>
    <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
```

```
46          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
          </graphEntries>
48      </callGraph>
        <customProperties key="priority">
50        <value xsi:type="am:StringObject" value="4" />
        </customProperties>
52      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="4" />
54      </customProperties>
    </tasks>
56  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
58      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
60      </graphEntries>
      </callGraph>
62    <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
64    </customProperties>
      <customProperties key="osekTaskGroup">
66      <value xsi:type="am:StringObject" value="3" />
      </customProperties>
68  </tasks>
    <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
70    <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
72        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
        </graphEntries>
74    </callGraph>
      <customProperties key="priority">
76      <value xsi:type="am:StringObject" value="2" />
      </customProperties>
78    <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
80    </customProperties>
    </tasks>
82  <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="
        non_preemptive" multipleTaskActivationLimit="2">
      <callGraph>
84      <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
86        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
        </graphEntries>
88    </callGraph>
      <customProperties key="priority">
90      <value xsi:type="am:StringObject" value="1" />
      </customProperties>
92    <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
94    </customProperties>
    </tasks>
96  <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
98      <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
100         <deviation>
```

```
            <lowerBound xsi:type="am:LongObject" value="5970000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="17970000" />
            <upperBound xsi:type="am:LongObject" value="18000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_3" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="11970000" />
            <upperBound xsi:type="am:LongObject" value="12000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="8970000" />
            <upperBound xsi:type="am:LongObject" value="9000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="17970000" />
            <upperBound xsi:type="am:LongObject" value="18000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
```

```
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23970000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="35977500" />
              <upperBound xsi:type="am:LongObject" value="36000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11992500" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
```

```
214             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
216       </structures>
      </structures>
218   </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
220   <defaultValue value="600.0" unit="MHz"/>
    </domains>
222 </hwModel>
  <osModel>
224   <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
226     <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
228   <osDataConsistency mode="noProtection" />
    </operatingSystems>
230 </osModel>
  <stimuliModel>
232   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
234   <recurrence value="80" unit="ms" />
    </stimuli>
236   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="0" unit="ms" />
238   <recurrence value="120" unit="ms" />
    </stimuli>
240   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
242   <recurrence value="160" unit="ms" />
    </stimuli>
244   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
      <offset value="0" unit="ms" />
246   <recurrence value="180" unit="ms" />
    </stimuli>
248   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
250   <recurrence value="200" unit="ms" />
    </stimuli>
252   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
      <offset value="0" unit="ms" />
254   <recurrence value="300" unit="ms" />
    </stimuli>
256   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
258   <recurrence value="1000" unit="ms" />
    </stimuli>
260 </stimuliModel>
  <constraintsModel />
262 <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
264   <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
266   <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
268   <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
270   <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2" entity="Runnable_2?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4" entity="Runnable_4?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6" entity="Runnable_6?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
        Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="" entity="
        Stimulus_Task_6?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.6: Variation 6 of Purely Periodic without Communication.

A.1.1.7. Variation 7

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
```

```
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="7" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="7" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_2_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="6" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="6" />
    </customProperties>
  </tasks>
  <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
```

```
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="
      non_preemptive" multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
        <calls xsi:type="am:SchedulePoint" />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5970000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="17970000" />
            <upperBound xsi:type="am:LongObject" value="18000000" />
            <distribution xsi:type="am:UniformDistribution" />
```

```
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_3" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="11970000" />
            <upperBound xsi:type="am:LongObject" value="12000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="8970000" />
            <upperBound xsi:type="am:LongObject" value="9000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="17970000" />
            <upperBound xsi:type="am:LongObject" value="18000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_6" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="23970000" />
            <upperBound xsi:type="am:LongObject" value="24000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_7_1" callback="false" service="false">
```

```
176       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
178           <value xsi:type="am:NeedDeviation">
                <deviation>
180               <lowerBound xsi:type="am:LongObject" value="35977500" />
                  <upperBound xsi:type="am:LongObject" value="36000000" />
182               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
184           </value>
            </default>
186       </runnableItems>
      </runnables>
188   <runnables name="Runnable_7_2" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
190       <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
192           <deviation>
                <lowerBound xsi:type="am:LongObject" value="11992500" />
194             <upperBound xsi:type="am:LongObject" value="12000000" />
                <distribution xsi:type="am:UniformDistribution" />
196           </deviation>
            </value>
198       </default>
        </runnableItems>
200   </runnables>
    </swModel>
202 <hwModel>
      <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
          IPC_1.0?type=HwFeature" puType="CPU"/>
204   <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
      </definitions>
206   <featureCategories name="Instructions" featureType="performance">
        <features name="IPC_1.0" value="1.0" />
208   </featureCategories>
      <structures name="System" structureType="System">
210     <structures name="Ecu_1" structureType="ECU">
          <structures name="Processor_1" structureType="Microcontroller">
212         <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
            </modules>
214         <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
              <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
216         </modules>
          </structures>
218     </structures>
      </structures>
220   <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
        <defaultValue value="600.0" unit="MHz"/>
222   </domains>
    </hwModel>
224 <osModel>
      <operatingSystems name="Generic_OS">
226     <taskSchedulers name="Scheduler_1">
          <schedulingAlgorithm xsi:type="am:OSEK" />
228     </taskSchedulers>
        <osDataConsistency mode="noProtection" />
230   </operatingSystems>
```

```
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="80" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="0" unit="ms" />
      <recurrence value="120" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
      <recurrence value="160" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
      <offset value="0" unit="ms" />
      <recurrence value="180" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="200" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
      <offset value="0" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
      <recurrence value="1000" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2" entity="Runnable_2?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4" entity="Runnable_4?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6" entity="Runnable_6?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
        Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
```

```
280     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
            PeriodicStimulus" />
282     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
            PeriodicStimulus" />
284     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="" entity="
            Stimulus_Task_6?type=PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
            PeriodicStimulus" />
286   </eventModel>
    <mappingModel addressMappingType="offset">
288     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
290     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
292     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
294     <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
296   </mappingModel>
    <componentsModel />
298 </am:Amalthea>
```

Listing A.7: Variation 7 of Purely Periodic without Communication.

A.1.1.8. Variation 8

```
    <?xml version="1.0" encoding="UTF-8"?>
2   <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
        " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
4       <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus"
            multipleTaskActivationLimit="2">
          <callGraph>
6           <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
8           </graphEntries>
          </callGraph>
10      </tasks>
        <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus"
            multipleTaskActivationLimit="2">
12        <callGraph>
            <graphEntries xsi:type="am:CallSequence" name="CallSequence_2_1">
14            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2?type=Runnable" />
            </graphEntries>
16        </callGraph>
        </tasks>
18      <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus"
            multipleTaskActivationLimit="2">
          <callGraph>
20          <graphEntries xsi:type="am:CallSequence" name="CallSequence_3_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
```

```
      </graphEntries>
    </callGraph>
  </tasks>
  <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_4_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4?type=Runnable" />
      </graphEntries>
    </callGraph>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_6_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6?type=Runnable" />
      </graphEntries>
    </callGraph>
  </tasks>
  <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_7_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
      </graphEntries>
    </callGraph>
  </tasks>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5970000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="17970000" />
            <upperBound xsi:type="am:LongObject" value="18000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
```

```
76          </value>
          </default>
78        </runnableItems>
      </runnables>
80      <runnables name="Runnable_3" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
82        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
84          <deviation>
              <lowerBound xsi:type="am:LongObject" value="11970000" />
86            <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
88          </deviation>
          </value>
90        </default>
        </runnableItems>
92    </runnables>
      <runnables name="Runnable_4" callback="false" service="false">
94      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
96        <value xsi:type="am:NeedDeviation">
            <deviation>
98            <lowerBound xsi:type="am:LongObject" value="8970000" />
              <upperBound xsi:type="am:LongObject" value="9000000" />
100           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
102       </value>
        </default>
104     </runnableItems>
      </runnables>
106   <runnables name="Runnable_5" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
108       <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
110         <deviation>
              <lowerBound xsi:type="am:LongObject" value="17970000" />
112           <upperBound xsi:type="am:LongObject" value="18000000" />
              <distribution xsi:type="am:UniformDistribution" />
114         </deviation>
          </value>
116       </default>
        </runnableItems>
118   </runnables>
      <runnables name="Runnable_6" callback="false" service="false">
120     <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
122       <value xsi:type="am:NeedDeviation">
            <deviation>
124           <lowerBound xsi:type="am:LongObject" value="23970000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
126           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
128       </value>
        </default>
130     </runnableItems>
      </runnables>
132   <runnables name="Runnable_7_1" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
```

```
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="35977500" />
            <upperBound xsi:type="am:LongObject" value="36000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_7_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="11992500" />
            <upperBound xsi:type="am:LongObject" value="12000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
</swModel>
<hwModel>
  <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
      IPC_1.0?type=HwFeature" puType="CPU"/>
  <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
  </definitions>
  <featureCategories name="Instructions" featureType="performance">
    <features name="IPC_1.0" value="1.0" />
  </featureCategories>
  <structures name="System" structureType="System">
    <structures name="Ecu_1" structureType="ECU">
      <structures name="Processor_1" structureType="Microcontroller">
        <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
        </modules>
        <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
          <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
        </modules>
      </structures>
    </structures>
  </structures>
  <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
    <defaultValue value="600.0" unit="MHz"/>
  </domains>
</hwModel>
<osModel>
  <operatingSystems name="Generic_OS">
    <taskSchedulers name="Scheduler_1">
      <schedulingAlgorithm xsi:type="am:EarliestDeadlineFirst" />
    </taskSchedulers>
    <osDataConsistency mode="noProtection" />
  </operatingSystems>
</osModel>
```

```
<stimuliModel>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
    <offset value="0" unit="ms" />
    <recurrence value="80" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
    <offset value="0" unit="ms" />
    <recurrence value="120" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
    <offset value="0" unit="ms" />
    <recurrence value="160" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
    <offset value="0" unit="ms" />
    <recurrence value="180" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
    <offset value="0" unit="ms" />
    <recurrence value="200" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
    <offset value="0" unit="ms" />
    <recurrence value="300" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
    <offset value="0" unit="ms" />
    <recurrence value="1000" unit="ms" />
  </stimuli>
</stimuliModel>
<constraintsModel>
  <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_1" severity="Critical"
      process="Task_1?type=Task">
    <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
      <limitValue value="10" unit="ms" />
    </limit>
  </requirements>
  <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_2" severity="Critical"
      process="Task_2?type=Task">
    <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
      <limitValue value="40" unit="ms" />
    </limit>
  </requirements>
  <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_3" severity="Critical"
      process="Task_3?type=Task">
    <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
      <limitValue value="60" unit="ms" />
    </limit>
  </requirements>
  <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_4" severity="Critical"
      process="Task_4?type=Task">
    <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
      <limitValue value="75" unit="ms" />
    </limit>
  </requirements>
  <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_5" severity="Critical"
      process="Task_5?type=Task">
    <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
```

```
242         <limitValue value="115" unit="ms" />
        </limit>
244     </requirements>
        <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_6" severity="Critical"
            process="Task_6?type=Task">
246       <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
          <limitValue value="300" unit="ms" />
248       </limit>
        </requirements>
250     <requirements xsi:type="am:ProcessRequirement" name="Deadline_Task_7" severity="Critical"
            process="Task_7?type=Task">
        <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
252       <limitValue value="960" unit="ms" />
        </limit>
254     </requirements>
      </constraintsModel>
256   <eventModel>
        <events xsi:type="am:ProcessEvent" name="Event_Task_1">
258       <entity xsi:type="am:Task" href="amlt:/#Task_1?type=Task" />
        </events>
260     <events xsi:type="am:ProcessEvent" name="Event_Task_2">
          <entity xsi:type="am:Task" href="amlt:/#Task_2?type=Task" />
262     </events>
        <events xsi:type="am:ProcessEvent" name="Event_Task_3">
264       <entity xsi:type="am:Task" href="amlt:/#Task_3?type=Task" />
        </events>
266     <events xsi:type="am:ProcessEvent" name="Event_Task_4">
          <entity xsi:type="am:Task" href="amlt:/#Task_4?type=Task" />
268     </events>
        <events xsi:type="am:ProcessEvent" name="Event_Task_5">
270       <entity xsi:type="am:Task" href="amlt:/#Task_5?type=Task" />
        </events>
272     <events xsi:type="am:ProcessEvent" name="Event_Task_6">
          <entity xsi:type="am:Task" href="amlt:/#Task_6?type=Task" />
274     </events>
        <events xsi:type="am:ProcessEvent" name="Event_Task_7">
276       <entity xsi:type="am:Task" href="amlt:/#Task_7?type=Task" />
        </events>
278     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1">
          <entity href="amlt:/#Runnable_1?type=Runnable" />
280     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_2">
282       <entity href="amlt:/#Runnable_2?type=Runnable" />
        </events>
284     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3">
          <entity href="amlt:/#Runnable_3?type=Runnable" />
286     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4">
288       <entity href="amlt:/#Runnable_4?type=Runnable" />
        </events>
290     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5">
          <entity href="amlt:/#Runnable_5?type=Runnable" />
292     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6">
294       <entity href="amlt:/#Runnable_6?type=Runnable" />
        </events>
296     <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1">
          <entity href="amlt:/#Runnable_7_1?type=Runnable" />
```

```
298     </events>
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2">
300       <entity href="amlt:/#Runnable_7_2?type=Runnable" />
        </events>
302     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_1?type=PeriodicStimulus" /
              >
304     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2">
306       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_2?type=PeriodicStimulus" /
              >
        </events>
308     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_3?type=PeriodicStimulus" /
              >
310     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4">
312       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_4?type=PeriodicStimulus" /
              >
        </events>
314     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_5?type=PeriodicStimulus" /
              >
316     </events>
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" description="">
318       <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_6?type=PeriodicStimulus" /
              >
        </events>
320     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7">
          <entity xsi:type="am:PeriodicStimulus" href="amlt:/#Stimulus_Task_7?type=PeriodicStimulus" /
              >
322     </events>
      </eventModel>
324   <mappingModel addressMappingType="offset">
        <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
326     <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
328     <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
330     <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
332     <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
      </mappingModel>
334   <componentsModel />
    </am:Amalthea>
```

Listing A.8: Variation 8 of Purely Periodic without Communication.

A.1.2. Client-Server without Reply

A.1.2.1. Variation 1

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="Stimulus_2?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="Content_2">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_2?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
            </items>
          </entries>
          <entries name="Content_3">
```

```
56        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_3?type=ModeLiteral"/>
58          </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_3">
60            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
            </items>
62        </entries>
          <entries name="Content_1">
64        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_1?type=ModeLiteral"/>
66          </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_1">
68            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
            </items>
70        </entries>
          <entries name="Content_4">
72        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_4?type=ModeLiteral"/>
74          </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_4">
76            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
            </items>
78        </entries>
          <defaultEntry>
80          <items xsi:type="am:CallSequence" name="CallSequence_2_default">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
82          </items>
          </defaultEntry>
84      </graphEntries>
      </callGraph>
86    <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
88    </customProperties>
      <customProperties key="osekTaskGroup">
90      <value xsi:type="am:StringObject" value="1" />
      </customProperties>
92  </tasks>
    <runnables name="Runnable_1_1" callback="false" service="false">
94    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
    </runnables>
96  <runnables name="Runnable_2_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
98      <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
100         <deviation>
              <lowerBound xsi:type="am:LongObject" value="594" />
102           <upperBound xsi:type="am:LongObject" value="600" />
              <distribution xsi:type="am:UniformDistribution" />
104         </deviation>
          </value>
106     </default>
      </runnableItems>
108 </runnables>
    <runnables name="Runnable_2_2" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
              <upperBound xsi:type="am:LongObject" value="30000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23760000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_default" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59" />
              <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_3" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
```

```
166     </runnables>
        <runnables name="Runnable_1_4" callback="false" service="false">
168       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
        </runnables>
170     <runnables name="Runnable_1_0" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
172     </runnables>
        <runnables name="Runnable_1" callback="false" service="false">
174       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
176           <value xsi:type="am:NeedDeviation">
                <deviation>
178               <lowerBound xsi:type="am:LongObject" value="5994000" />
                  <upperBound xsi:type="am:LongObject" value="6000000" />
180               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
182           </value>
            </default>
184       </runnableItems>
        </runnables>
186     <modes name="MessageMode">
          <literals name="MessageContent_0">
188         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
190         </customProperties>
          </literals>
192       <literals name="MessageContent_1">
            <customProperties key="enumValue">
194           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
196       </literals>
          <literals name="MessageContent_2">
198         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
200         </customProperties>
          </literals>
202       <literals name="MessageContent_3">
            <customProperties key="enumValue">
204           <value xsi:type="am:LongObject" value="3" />
            </customProperties>
206       </literals>
          <literals name="MessageContent_4">
208         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="4" />
210         </customProperties>
          </literals>
212     </modes>
        <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
214       <size value="8" unit="bit" />
        </modeLabels>
216   </swModel>
      <hwModel>
218     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
220     </definitions>
```

```
    <featureCategories name="Instructions" featureType="performance">
222   <features name="IPC_1.0" value="1.0" />
    </featureCategories>
224 <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
226     <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
228       </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
230         <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
232     </structures>
      </structures>
234 </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
236   <defaultValue value="600.0" unit="MHz"/>
    </domains>
238 </hwModel>
    <osModel>
240   <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
242     <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
244   <osDataConsistency mode="noProtection" />
    </operatingSystems>
246 </osModel>
    <stimuliModel>
248   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_2">
      <offset value="1" unit="ms" />
250   <recurrence value="50" unit="ms" />
    </stimuli>
252 <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
      <offset value="0" unit="ms" />
254   <recurrence value="100" unit="ms" />
    </stimuli>
256 </stimuliModel>
    <constraintsModel />
258 <eventModel>
      <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
260   <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
          />
262   <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
          Runnable" />
264   <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
          Runnable" />
266   <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
          Runnable" />
268   <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
          Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
        ?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" entity="Stimulus_2?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.9: Variation 1 of Client-Server without Reply.

A.1.2.2. Variation 2

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
```

```
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="Stimulus_2?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries name="Content_2">
      <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                MessageMode/MessageContent_2?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
          </items>
        </entries>
        <entries name="Content_3">
      <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                MessageMode/MessageContent_3?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
          </items>
        </entries>
        <entries name="Content_1">
      <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                MessageMode/MessageContent_1?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
          </items>
        </entries>
        <entries name="Content_4">
      <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                MessageMode/MessageContent_4?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
          </items>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_2_default">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
```

```
82            </items>
            </defaultEntry>
84        </graphEntries>
      </callGraph>
86    <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
88    </customProperties>
      <customProperties key="osekTaskGroup">
90      <value xsi:type="am:StringObject" value="1" />
      </customProperties>
92  </tasks>
    <runnables name="Runnable_1_1" callback="false" service="false">
94    <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          request" />
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
96    <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          release" />
    </runnables>
98  <runnables name="Runnable_2_1" callback="false" service="false">
      <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          request" />
100   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
102       <value xsi:type="am:NeedDeviation">
            <deviation>
104           <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="600" />
106           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
108       </value>
        </default>
110   </runnableItems>
      <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          release" />
112 </runnables>
    <runnables name="Runnable_2_2" callback="false" service="false">
114   <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          request" />
      <runnableItems xsi:type="am:ExecutionNeed">
116     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
118         <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
120           <upperBound xsi:type="am:LongObject" value="30000" />
              <distribution xsi:type="am:UniformDistribution" />
122         </deviation>
          </value>
124     </default>
      </runnableItems>
126   <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          release" />
    </runnables>
128 <runnables name="Runnable_2_3" callback="false" service="false">
      <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          request" />
130   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
```

```
132           <value xsi:type="am:NeedDeviation">
                <deviation>
134               <lowerBound xsi:type="am:LongObject" value="594000" />
                  <upperBound xsi:type="am:LongObject" value="600000" />
136               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
138           </value>
            </default>
140       </runnableItems>
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
142     </runnables>
        <runnables name="Runnable_2_4" callback="false" service="false">
144       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
          <runnableItems xsi:type="am:ExecutionNeed">
146         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
148             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="23760000" />
150               <upperBound xsi:type="am:LongObject" value="24000000" />
                  <distribution xsi:type="am:UniformDistribution" />
152             </deviation>
              </value>
154         </default>
          </runnableItems>
156       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
        </runnables>
158     <runnables name="Runnable_2_default" callback="false" service="false">
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
160       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
162           <value xsi:type="am:NeedDeviation">
                <deviation>
164               <lowerBound xsi:type="am:LongObject" value="59" />
                  <upperBound xsi:type="am:LongObject" value="60" />
166               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
168           </value>
            </default>
170       </runnableItems>
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
172     </runnables>
        <runnables name="Runnable_1_2" callback="false" service="false">
174       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
176       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
        </runnables>
178     <runnables name="Runnable_1_3" callback="false" service="false">
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
```

```
180       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
182     </runnables>
        <runnables name="Runnable_1_4" callback="false" service="false">
184       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
186       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
        </runnables>
188     <runnables name="Runnable_1_0" callback="false" service="false">
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
190       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
192     </runnables>
        <runnables name="Runnable_1" callback="false" service="false">
194       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
196           <value xsi:type="am:NeedDeviation">
                <deviation>
198               <lowerBound xsi:type="am:LongObject" value="5994000" />
                  <upperBound xsi:type="am:LongObject" value="6000000" />
200               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
202           </value>
            </default>
204       </runnableItems>
        </runnables>
206     <modes name="MessageMode">
          <literals name="MessageContent_0">
208         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
210         </customProperties>
          </literals>
212       <literals name="MessageContent_1">
            <customProperties key="enumValue">
214           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
216       </literals>
          <literals name="MessageContent_2">
218         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
220         </customProperties>
          </literals>
222       <literals name="MessageContent_3">
            <customProperties key="enumValue">
224           <value xsi:type="am:LongObject" value="3" />
            </customProperties>
226       </literals>
          <literals name="MessageContent_4">
228         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="4" />
```

```
230         </customProperties>
        </literals>
232     </modes>
      <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
234       <size value="8" unit="bit" />
      </modeLabels>
236   </swModel>
    <hwModel>
238     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
          IPC_1.0?type=HwFeature" puType="CPU"/>
      <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
240     </definitions>
      <featureCategories name="Instructions" featureType="performance">
242       <features name="IPC_1.0" value="1.0" />
      </featureCategories>
244     <structures name="System" structureType="System">
        <structures name="Ecu_1" structureType="ECU">
246       <structures name="Processor_1" structureType="Microcontroller">
            <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
248         </modules>
            <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
250           <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
            </modules>
252       </structures>
        </structures>
254     </structures>
      <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
256       <defaultValue value="600.0" unit="MHz"/>
      </domains>
258   </hwModel>
    <osModel>
260     <semaphores name="Semaphore" initialValue="0" maxValue="1" priorityCeilingProtocol="true" />
      <operatingSystems name="Generic_OS">
262       <taskSchedulers name="Scheduler_1">
          <schedulingAlgorithm xsi:type="am:OSEK" />
264       </taskSchedulers>
        <osDataConsistency mode="noProtection" />
266     </operatingSystems>
    </osModel>
268   <stimuliModel>
      <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_2">
270       <offset value="1" unit="ms" />
        <recurrence value="50" unit="ms" />
272     </stimuli>
      <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
274       <offset value="0" unit="ms" />
        <recurrence value="100" unit="ms" />
276     </stimuli>
    </stimuliModel>
278   <constraintsModel />
    <eventModel>
280     <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
      <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
282     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
          />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
        ?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" entity="Stimulus_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:SemaphoreEvent" name="Event_Semaphore" entity="Semaphore?type=Semaphore"
        />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.10: Variation 2 of Client-Server without Reply.

A.1.2.3. Variation 3

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
```

```
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_2?type=InterProcessStimulus"
               />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="Content_2">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_2?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
            </items>
          </entries>
          <entries name="Content_3">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_3?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
            </items>
          </entries>
          <entries name="Content_1">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_1?type=ModeLiteral"/>
```

```
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
          </items>
        </entries>
        <entries name="Content_4">
      <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                MessageMode/MessageContent_4?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
          </items>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_2_default">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
          </items>
        </defaultEntry>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_2_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594" />
            <upperBound xsi:type="am:LongObject" value="600" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700" />
            <upperBound xsi:type="am:LongObject" value="30000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
```

```
    <runnables name="Runnable_2_3" callback="false" service="false">
124   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
126       <value xsi:type="am:NeedDeviation">
            <deviation>
128           <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
130           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
132       </value>
        </default>
134   </runnableItems>
    </runnables>
136 <runnables name="Runnable_2_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
138     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
140         <deviation>
              <lowerBound xsi:type="am:LongObject" value="23760000" />
142           <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
144         </deviation>
          </value>
146     </default>
      </runnableItems>
148 </runnables>
    <runnables name="Runnable_2_default" callback="false" service="false">
150   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
152       <value xsi:type="am:NeedDeviation">
            <deviation>
154           <lowerBound xsi:type="am:LongObject" value="59" />
              <upperBound xsi:type="am:LongObject" value="60" />
156           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
158       </value>
        </default>
160   </runnableItems>
    </runnables>
162 <runnables name="Runnable_1_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
164 </runnables>
    <runnables name="Runnable_1_3" callback="false" service="false">
166   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
    </runnables>
168 <runnables name="Runnable_1_4" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
170 </runnables>
    <runnables name="Runnable_1_0" callback="false" service="false">
172   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
    </runnables>
174 <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
176     <default key="Instructions">
```

```
          <value xsi:type="am:NeedDeviation">
178         <deviation>
              <lowerBound xsi:type="am:LongObject" value="5994000" />
180           <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
182         </deviation>
          </value>
184     </default>
      </runnableItems>
186   </runnables>
    <modes name="MessageMode">
188   <literals name="MessageContent_0">
        <customProperties key="enumValue">
190       <value xsi:type="am:LongObject" value="0" />
        </customProperties>
192   </literals>
      <literals name="MessageContent_1">
194     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
196     </customProperties>
      </literals>
198   <literals name="MessageContent_2">
        <customProperties key="enumValue">
200       <value xsi:type="am:LongObject" value="2" />
        </customProperties>
202   </literals>
      <literals name="MessageContent_3">
204     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="3" />
206     </customProperties>
      </literals>
208   <literals name="MessageContent_4">
        <customProperties key="enumValue">
210       <value xsi:type="am:LongObject" value="4" />
        </customProperties>
212   </literals>
    </modes>
214 <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
      <size value="8" unit="bit" />
216 </modeLabels>
  </swModel>
218 <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
220 <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
222 <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
224 </featureCategories>
    <structures name="System" structureType="System">
226   <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
228       <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
230       <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
```

```
232           </modules>
            </structures>
234       </structures>
      </structures>
236     <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
        <defaultValue value="600.0" unit="MHz"/>
238     </domains>
    </hwModel>
240   <osModel>
      <operatingSystems name="Generic_OS">
242       <taskSchedulers name="Scheduler_1">
          <schedulingAlgorithm xsi:type="am:OSEK" />
244       </taskSchedulers>
        <osDataConsistency mode="noProtection" />
246     </operatingSystems>
    </osModel>
248   <stimuliModel>
      <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
250       <offset value="0" unit="ms" />
        <recurrence value="100" unit="ms" />
252     </stimuli>
      <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_2" />
254   </stimuliModel>
    <constraintsModel />
256   <eventModel>
      <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
258     <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
          />
260     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
          Runnable" />
262     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
          Runnable" />
264     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
          Runnable" />
266     <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
          Runnable" />
268     <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
          ?type=Runnable" />
270     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
          PeriodicStimulus" />
      <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" />
272   </eventModel>
    <mappingModel addressMappingType="offset">
274     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
      <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
276     <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
          ProcessingUnit" />
```

```
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.11: Variation 3 of Client-Server without Reply.

A.1.2.4. Variation 4

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_2?type=InterProcessStimulus"
              />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
```

```
44      </tasks>
        <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
46        <callGraph>
            <graphEntries xsi:type="am:ModeSwitch">
48            <entries name="Content_2">
            <condition>
50                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      MessageMode/MessageContent_2?type=ModeLiteral"/>
              </condition>
52            <items xsi:type="am:CallSequence" name="CallSequence_2_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
54            </items>
            </entries>
56          <entries name="Content_3">
            <condition>
58                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      MessageMode/MessageContent_3?type=ModeLiteral"/>
              </condition>
60            <items xsi:type="am:CallSequence" name="CallSequence_2_3">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
62            </items>
            </entries>
64          <entries name="Content_1">
            <condition>
66                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      MessageMode/MessageContent_1?type=ModeLiteral"/>
              </condition>
68            <items xsi:type="am:CallSequence" name="CallSequence_2_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
70            </items>
            </entries>
72          <entries name="Content_4">
            <condition>
74                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      MessageMode/MessageContent_4?type=ModeLiteral"/>
              </condition>
76            <items xsi:type="am:CallSequence" name="CallSequence_2_4">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
78            </items>
            </entries>
80          <defaultEntry>
              <items xsi:type="am:CallSequence" name="CallSequence_2_default">
82              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
              </items>
84          </defaultEntry>
          </graphEntries>
86      </callGraph>
        <customProperties key="priority">
88        <value xsi:type="am:StringObject" value="2" />
        </customProperties>
90      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="2" />
92      </customProperties>
      </tasks>
94    <runnables name="Runnable_1_1" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
96    </runnables>
```

```
    <runnables name="Runnable_2_1" callback="false" service="false">
98    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
100       <value xsi:type="am:NeedDeviation">
            <deviation>
102           <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="600" />
104           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
106       </value>
        </default>
108   </runnableItems>
    </runnables>
110 <runnables name="Runnable_2_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
112     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
114         <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
116           <upperBound xsi:type="am:LongObject" value="30000" />
              <distribution xsi:type="am:UniformDistribution" />
118         </deviation>
          </value>
120     </default>
      </runnableItems>
122 </runnables>
    <runnables name="Runnable_2_3" callback="false" service="false">
124   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
126       <value xsi:type="am:NeedDeviation">
            <deviation>
128           <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
130           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
132       </value>
        </default>
134   </runnableItems>
    </runnables>
136 <runnables name="Runnable_2_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
138     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
140         <deviation>
              <lowerBound xsi:type="am:LongObject" value="23760000" />
142           <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
144         </deviation>
          </value>
146     </default>
      </runnableItems>
148 </runnables>
    <runnables name="Runnable_2_default" callback="false" service="false">
150   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
152       <value xsi:type="am:NeedDeviation">
            <deviation>
154           <lowerBound xsi:type="am:LongObject" value="59" />
```

```
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_3" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_4" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5994000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <modes name="MessageMode">
    <literals name="MessageContent_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="MessageContent_1">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="1" />
      </customProperties>
    </literals>
    <literals name="MessageContent_2">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="2" />
      </customProperties>
    </literals>
    <literals name="MessageContent_3">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="3" />
      </customProperties>
    </literals>
    <literals name="MessageContent_4">
```

```
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="4" />
      </customProperties>
    </literals>
  </modes>
  <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
    <size value="8" unit="bit" />
  </modeLabels>
</swModel>
<hwModel>
  <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
      IPC_1.0?type=HwFeature" puType="CPU"/>
  <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
  </definitions>
  <featureCategories name="Instructions" featureType="performance">
    <features name="IPC_1.0" value="1.0" />
  </featureCategories>
  <structures name="System" structureType="System">
    <structures name="Ecu_1" structureType="ECU">
      <structures name="Processor_1" structureType="Microcontroller">
        <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
        </modules>
        <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
          <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
        </modules>
      </structures>
    </structures>
  </structures>
  <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
    <defaultValue value="600.0" unit="MHz"/>
  </domains>
</hwModel>
<osModel>
  <operatingSystems name="Generic_OS">
    <taskSchedulers name="Scheduler_1">
      <schedulingAlgorithm xsi:type="am:OSEK" />
    </taskSchedulers>
    <osDataConsistency mode="noProtection" />
  </operatingSystems>
</osModel>
<stimuliModel>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
    <offset value="0" unit="ms" />
    <recurrence value="100" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_2" />
</stimuliModel>
<constraintsModel />
<eventModel>
  <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
  <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
      />
  <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
      Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
        ?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.12: Variation 4 of Client-Server without Reply.

A.1.2.5. Variation 5

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
```

```
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_2?type=InterProcessStimulus"
               />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="Content_2">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_2?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
            </items>
          </entries>
          <entries name="Content_3">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_3?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
            </items>
          </entries>
          <entries name="Content_1">
        <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  MessageMode/MessageContent_1?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_2_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
            </items>
          </entries>
```

```
72          <entries name="Content_4">
          <condition>
74              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_4?type=ModeLiteral"/>
            </condition>
76          <items xsi:type="am:CallSequence" name="CallSequence_2_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
78          </items>
          </entries>
80        <defaultEntry>
            <items xsi:type="am:CallSequence" name="CallSequence_2_default">
82            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
            </items>
84        </defaultEntry>
        </graphEntries>
86    </callGraph>
      <customProperties key="priority">
88      <value xsi:type="am:StringObject" value="2" />
      </customProperties>
90    <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
92    </customProperties>
  </tasks>
94  <runnables name="Runnable_1_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
96  </runnables>
    <runnables name="Runnable_2_1" callback="false" service="false">
98    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
100       <value xsi:type="am:NeedDeviation">
            <deviation>
102           <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="600" />
104           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
106       </value>
        </default>
108   </runnableItems>
  </runnables>
110 <runnables name="Runnable_2_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
112     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
114         <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
116           <upperBound xsi:type="am:LongObject" value="30000" />
              <distribution xsi:type="am:UniformDistribution" />
118         </deviation>
          </value>
120     </default>
      </runnableItems>
122 </runnables>
    <runnables name="Runnable_2_3" callback="false" service="false">
124   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
126       <value xsi:type="am:NeedDeviation">
            <deviation>
```

```
            <lowerBound xsi:type="am:LongObject" value="594000" />
            <upperBound xsi:type="am:LongObject" value="600000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="23760000" />
            <upperBound xsi:type="am:LongObject" value="24000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2_default" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59" />
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_3" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_4" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5994000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
```

```
182             </deviation>
              </value>
184           </default>
        </runnableItems>
186     </runnables>
      <modes name="MessageMode">
188       <literals name="MessageContent_0">
          <customProperties key="enumValue">
190         <value xsi:type="am:LongObject" value="0" />
          </customProperties>
192       </literals>
        <literals name="MessageContent_1">
194       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="1" />
196       </customProperties>
        </literals>
198     <literals name="MessageContent_2">
          <customProperties key="enumValue">
200         <value xsi:type="am:LongObject" value="2" />
          </customProperties>
202     </literals>
        <literals name="MessageContent_3">
204       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="3" />
206       </customProperties>
        </literals>
208     <literals name="MessageContent_4">
          <customProperties key="enumValue">
210         <value xsi:type="am:LongObject" value="4" />
          </customProperties>
212     </literals>
      </modes>
214   <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
        <size value="8" unit="bit" />
216   </modeLabels>
    </swModel>
218 <hwModel>
      <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
          IPC_1.0?type=HwFeature" puType="CPU"/>
220   <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
      </definitions>
222   <featureCategories name="Instructions" featureType="performance">
        <features name="IPC_1.0" value="1.0" />
224   </featureCategories>
      <structures name="System" structureType="System">
226     <structures name="Ecu_1" structureType="ECU">
          <structures name="Processor_1" structureType="Microcontroller">
228         <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
            </modules>
230         <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
              <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
232         </modules>
          </structures>
234     </structures>
      </structures>
236   <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
```

```
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
      <offset value="0" unit="ms" />
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
        ?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.13: Variation 5 of Client-Server without Reply.

A.1.2.6. Variation 6

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_2?type=InterProcessStimulus"
              />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
```

```
48          <entries name="Content_2">
        <condition>
50              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_2?type=ModeLiteral"/>
            </condition>
52          <items xsi:type="am:CallSequence" name="CallSequence_2_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
54          </items>
          </entries>
56        <entries name="Content_3">
        <condition>
58              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_3?type=ModeLiteral"/>
            </condition>
60          <items xsi:type="am:CallSequence" name="CallSequence_2_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
62          </items>
          </entries>
64        <entries name="Content_1">
        <condition>
66              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_1?type=ModeLiteral"/>
            </condition>
68          <items xsi:type="am:CallSequence" name="CallSequence_2_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
70          </items>
          </entries>
72        <entries name="Content_4">
        <condition>
74              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_4?type=ModeLiteral"/>
            </condition>
76          <items xsi:type="am:CallSequence" name="CallSequence_2_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
78          </items>
          </entries>
80        <defaultEntry>
            <items xsi:type="am:CallSequence" name="CallSequence_2_default">
82            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
            </items>
84        </defaultEntry>
        </graphEntries>
86    </callGraph>
      <customProperties key="priority">
88      <value xsi:type="am:StringObject" value="2" />
      </customProperties>
90    <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
92    </customProperties>
  </tasks>
94  <runnables name="Runnable_1_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
96  </runnables>
    <runnables name="Runnable_2_1" callback="false" service="false">
98    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
100       <value xsi:type="am:NeedDeviation">
```

```
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="606" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
              <upperBound xsi:type="am:LongObject" value="30300" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="606000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23760000" />
              <upperBound xsi:type="am:LongObject" value="24240000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_default" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="58" />
              <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
```

```
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_3" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_4" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5994000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <modes name="MessageMode">
      <literals name="MessageContent_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageContent_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="MessageContent_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
      <literals name="MessageContent_3">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="3" />
        </customProperties>
      </literals>
      <literals name="MessageContent_4">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="4" />
        </customProperties>
      </literals>
```

```
    </modes>
    <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
      <offset value="0" unit="ms" />
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
        Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
        ?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.14: Variation 6 of Client-Server without Reply.

A.1.2.7. Variation 7

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_4?type=Runnable" />
            </items>
          </entries>
```

```
22          <entries probability="20.0">
              <items xsi:type="am:CallSequence" name="CallSequence_1_1">
24              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
              </items>
26          </entries>
            <entries probability="15.0">
28            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
30            </items>
            </entries>
32        </graphEntries>
          <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
34          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_2?type=InterProcessStimulus"
                 />
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
36        </graphEntries>
        </callGraph>
38      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
40      </customProperties>
        <customProperties key="osekTaskGroup">
42        <value xsi:type="am:StringObject" value="1" />
        </customProperties>
44    </tasks>
      <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="2">
46      <callGraph>
          <graphEntries xsi:type="am:ModeSwitch">
48          <entries name="Content_2">
          <condition>
50              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_2?type=ModeLiteral"/>
            </condition>
52          <items xsi:type="am:CallSequence" name="CallSequence_2_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_2?type=Runnable" />
54          </items>
          </entries>
56        <entries name="Content_3">
          <condition>
58              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_3?type=ModeLiteral"/>
            </condition>
60          <items xsi:type="am:CallSequence" name="CallSequence_2_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_3?type=Runnable" />
62          </items>
          </entries>
64        <entries name="Content_1">
          <condition>
66              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_1?type=ModeLiteral"/>
            </condition>
68          <items xsi:type="am:CallSequence" name="CallSequence_2_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_1?type=Runnable" />
70          </items>
          </entries>
72        <entries name="Content_4">
          <condition>
74              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    MessageMode/MessageContent_4?type=ModeLiteral"/>
```

```
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_2_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_4?type=Runnable" />
          </items>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_2_default">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_default?type=Runnable" />
          </items>
        </defaultEntry>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="MessageMode/MessageContent_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_2_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594" />
            <upperBound xsi:type="am:LongObject" value="606" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700" />
            <upperBound xsi:type="am:LongObject" value="30300" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_2_3" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594000" />
            <upperBound xsi:type="am:LongObject" value="606000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
```

```
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23760000" />
              <upperBound xsi:type="am:LongObject" value="24240000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_2_default" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="58" />
              <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_3" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_3?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_4" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_4?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="MessageMode/MessageContent_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5994000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
```

```
    </runnables>
    <modes name="MessageMode">
      <literals name="MessageContent_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageContent_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="MessageContent_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
      <literals name="MessageContent_3">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="3" />
        </customProperties>
      </literals>
      <literals name="MessageContent_4">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="4" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="message" initialValue="MessageMode/MessageContent_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
```

```
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_1">
      <offset value="0" unit="ms" />
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_2" entity="Runnable_1_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_3" entity="Runnable_1_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_4" entity="Runnable_1_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_1" entity="Runnable_2_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_2" entity="Runnable_2_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_3" entity="Runnable_2_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_4" entity="Runnable_2_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_default" entity="Runnable_2_default
        ?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_1" entity="Stimulus_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.15: Variation 7 of Client-Server without Reply.

A.1.3. State Machine

A.1.3.1. Variation 1

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
          <entries probability="25.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="State_1">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_1?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
          <entries name="State_0">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_0?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
```

```
50          <entries name="State_2">
              <condition>
52              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_2?type=ModeLiteral"/>
              </condition>
54            <items xsi:type="am:CallSequence" name="CallSequence_State_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
56            </items>
              <items xsi:type="am:ModeSwitch" />
58          </entries>
          </graphEntries>
60      </callGraph>
        <customProperties key="priority">
62        <value xsi:type="am:StringObject" value="1" />
        </customProperties>
64      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="1" />
66      </customProperties>
      </tasks>
68    <runnables name="Runnable_1_1" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/MessageContent_1?type=ModeLiteral" />
70    </runnables>
      <runnables name="Runnable_State_0" callback="false" service="false">
72      <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
74          <value xsi:type="am:NeedDeviation">
              <deviation>
76              <lowerBound xsi:type="am:LongObject" value="59" />
                <upperBound xsi:type="am:LongObject" value="60" />
78              <distribution xsi:type="am:UniformDistribution" />
              </deviation>
80          </value>
          </default>
82      </runnableItems>
      </runnables>
84    <runnables name="Runnable_State_1" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
86        <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
88            <deviation>
                <lowerBound xsi:type="am:LongObject" value="59400" />
90              <upperBound xsi:type="am:LongObject" value="60000" />
                <distribution xsi:type="am:UniformDistribution" />
92            </deviation>
            </value>
94        </default>
        </runnableItems>
96    </runnables>
      <runnables name="Runnable_State_2" callback="false" service="false">
98      <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
100         <value xsi:type="am:NeedDeviation">
              <deviation>
102             <lowerBound xsi:type="am:LongObject" value="29700000" />
                <upperBound xsi:type="am:LongObject" value="30000000" />
104             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
```

```
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/MessageContent_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_2?type=ModeLiteral" />
  </runnables>
  <modes name="State">
    <literals name="State_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="State_1">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="1" />
      </customProperties>
    </literals>
    <literals name="State_2">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="2" />
      </customProperties>
    </literals>
  </modes>
  <modes name="Message">
    <literals name="MessageContent_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="MessageContent_1">
      <customProperties key="enumValue">
```

```
160           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
162       </literals>
        </modes>
164     <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
          <size value="8" unit="bit" />
166     </modeLabels>
        <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
168       <size value="1" unit="bit" />
        </modeLabels>
170   </swModel>
      <hwModel>
172     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
174     </definitions>
        <featureCategories name="Instructions" featureType="performance">
176       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
178     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
180         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
182           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
184             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
186         </structures>
          </structures>
188     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
190       <defaultValue value="600.0" unit="MHz"/>
        </domains>
192   </hwModel>
      <osModel>
194     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
196         <schedulingAlgorithm xsi:type="am:OSEK" />
          </taskSchedulers>
198       <osDataConsistency mode="noProtection" />
        </operatingSystems>
200   </osModel>
      <stimuliModel>
202     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
          <offset value="0" unit="ms" />
204       <recurrence value="100" unit="ms" />
        </stimuli>
206     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
          <offset value="15" unit="ms" />
208       <recurrence value="60" unit="ms" />
        </stimuli>
210   </stimuliModel>
      <constraintsModel />
212   <eventModel>
        <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
214     <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.16: Variation 1 of State Machine.

A.1.3.2. Variation 2

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
          <entries probability="25.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
```

```
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries name="State_1">
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_1?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_State_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch" />
        </entries>
        <entries name="State_0">
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_0?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch" />
        </entries>
        <entries name="State_2">
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_2?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_State_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch" />
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1_1" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          request" />
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_1?type=ModeLiteral" />
      <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
          release" />
    </runnables>
    <runnables name="Runnable_State_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59" />
              <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_State_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400" />
              <upperBound xsi:type="am:LongObject" value="60000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_State_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
              <upperBound xsi:type="am:LongObject" value="30000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
```

```
124     </runnables>
        <runnables name="Runnable_1_0" callback="false" service="false">
126       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/MessageContent_0?type=ModeLiteral" />
128       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
        </runnables>
130     <runnables name="Runnable_Transition_0" callback="false" service="false">
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
132       <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_0?type=ModeLiteral" />
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
134     </runnables>
        <runnables name="Runnable_Transition_1" callback="false" service="false">
136       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_1?type=ModeLiteral" />
138       <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
        </runnables>
140     <runnables name="Runnable_Transition_2" callback="false" service="false">
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              request" />
142       <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_2?type=ModeLiteral" />
          <runnableItems xsi:type="am:SemaphoreAccess" semaphore="Semaphore?type=Semaphore" access="
              release" />
144     </runnables>
        <modes name="State">
146       <literals name="State_0">
            <customProperties key="enumValue">
148           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
150       </literals>
          <literals name="State_1">
152         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
154         </customProperties>
          </literals>
156       <literals name="State_2">
            <customProperties key="enumValue">
158           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
160       </literals>
        </modes>
162     <modes name="Message">
          <literals name="MessageContent_0">
164         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
166         </customProperties>
          </literals>
168       <literals name="MessageContent_1">
            <customProperties key="enumValue">
```

```
170           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
172       </literals>
        </modes>
174     <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
          <size value="8" unit="bit" />
176     </modeLabels>
        <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
178       <size value="1" unit="bit" />
        </modeLabels>
180   </swModel>
      <hwModel>
182     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
184     </definitions>
        <featureCategories name="Instructions" featureType="performance">
186       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
188     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
190         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
192           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
194             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
196         </structures>
          </structures>
198     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
200       <defaultValue value="600.0" unit="MHz"/>
        </domains>
202   </hwModel>
      <osModel>
204     <semaphores name="Semaphore" initialValue="0" maxValue="1" priorityCeilingProtocol="true" />
        <operatingSystems name="Generic_OS">
206       <taskSchedulers name="Scheduler_1">
            <schedulingAlgorithm xsi:type="am:OSEK" />
208       </taskSchedulers>
          <osDataConsistency mode="noProtection" />
210     </operatingSystems>
      </osModel>
212   <stimuliModel>
        <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
214       <offset value="0" unit="ms" />
          <recurrence value="100" unit="ms" />
216     </stimuli>
        <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
218       <offset value="15" unit="ms" />
          <recurrence value="60" unit="ms" />
220     </stimuli>
      </stimuliModel>
222   <constraintsModel />
      <eventModel>
224     <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
```

```
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:SemaphoreEvent" name="Event_Semaphore" entity="Semaphore?type=Semaphore"
        />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.17: Variation 2 of State Machine.

A.1.3.3. Variation 3

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
```

```
12          <entries probability="25.0">
              <items xsi:type="am:CallSequence" name="CallSequence_1_1">
14              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
              </items>
16          </entries>
          </graphEntries>
18        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
20        </graphEntries>
        </callGraph>
22      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
24      </customProperties>
        <customProperties key="osekTaskGroup">
26        <value xsi:type="am:StringObject" value="1" />
        </customProperties>
28    </tasks>
      <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
30      <callGraph>
          <graphEntries xsi:type="am:ModeSwitch">
32          <entries name="State_1">
              <condition>
34              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_1?type=ModeLiteral"/>
              </condition>
36            <items xsi:type="am:CallSequence" name="CallSequence_State_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
38            </items>
              <items xsi:type="am:ModeSwitch" />
40          </entries>
            <entries name="State_0">
42            <condition>
                <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_0?type=ModeLiteral"/>
44            </condition>
              <items xsi:type="am:CallSequence" name="CallSequence_State_0">
46              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
              </items>
48            <items xsi:type="am:ModeSwitch" />
            </entries>
50          <entries name="State_2">
              <condition>
52              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_2?type=ModeLiteral"/>
              </condition>
54            <items xsi:type="am:CallSequence" name="CallSequence_State_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
56            </items>
              <items xsi:type="am:ModeSwitch" />
58          </entries>
          </graphEntries>
60      </callGraph>
        <customProperties key="priority">
62        <value xsi:type="am:StringObject" value="2" />
        </customProperties>
64      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="2" />
```

```
66      </customProperties>
    </tasks>
68  <runnables name="Runnable_1_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_1?type=ModeLiteral" />
70  </runnables>
    <runnables name="Runnable_State_0" callback="false" service="false">
72    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
74        <value xsi:type="am:NeedDeviation">
            <deviation>
76            <lowerBound xsi:type="am:LongObject" value="59" />
              <upperBound xsi:type="am:LongObject" value="60" />
78            <distribution xsi:type="am:UniformDistribution" />
            </deviation>
80        </value>
        </default>
82    </runnableItems>
    </runnables>
84  <runnables name="Runnable_State_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
86      <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
88          <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400" />
90            <upperBound xsi:type="am:LongObject" value="60000" />
              <distribution xsi:type="am:UniformDistribution" />
92          </deviation>
          </value>
94      </default>
      </runnableItems>
96  </runnables>
    <runnables name="Runnable_State_2" callback="false" service="false">
98    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
100       <value xsi:type="am:NeedDeviation">
            <deviation>
102           <lowerBound xsi:type="am:LongObject" value="29700000" />
              <upperBound xsi:type="am:LongObject" value="30000000" />
104           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
106       </value>
        </default>
108   </runnableItems>
    </runnables>
110 <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
112     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
114         <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
116           <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
118         </deviation>
          </value>
120     </default>
      </runnableItems>
122 </runnables>
```

```
    <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
    <modes name="State">
      <literals name="State_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="State_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="State_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
    <modes name="Message">
      <literals name="MessageContent_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageContent_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
```

```
176       <features name="IPC_1.0" value="1.0" />
      </featureCategories>
178     <structures name="System" structureType="System">
        <structures name="Ecu_1" structureType="ECU">
180       <structures name="Processor_1" structureType="Microcontroller">
            <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
182         </modules>
            <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
184           <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
            </modules>
186       </structures>
        </structures>
188     </structures>
      <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
190     <defaultValue value="600.0" unit="MHz"/>
      </domains>
192   </hwModel>
    <osModel>
194   <operatingSystems name="Generic_OS">
        <taskSchedulers name="Scheduler_1">
196       <schedulingAlgorithm xsi:type="am:OSEK" />
        </taskSchedulers>
198     <osDataConsistency mode="noProtection" />
      </operatingSystems>
200 </osModel>
    <stimuliModel>
202   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
        <offset value="0" unit="ms" />
204     <recurrence value="100" unit="ms" />
      </stimuli>
206   <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
        <offset value="15" unit="ms" />
208     <recurrence value="60" unit="ms" />
      </stimuli>
210 </stimuliModel>
    <constraintsModel />
212 <eventModel>
      <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
214   <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
          />
216   <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
          Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
          Runnable" />
218   <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
          type=Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
          type=Runnable" />
220   <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
          type=Runnable" />
      <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
          Runnable_Transition_0?type=Runnable" />
222   <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
          Runnable_Transition_1?type=Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.18: Variation 3 of State Machine.

A.1.3.4. Variation 4

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
          <entries probability="25.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_Task_2?type=
              InterProcessStimulus" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
```

```
    <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="State_1">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_1?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
          <entries name="State_0">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_0?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
          <entries name="State_2">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_2?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <runnables name="Runnable_1_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_State_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59" />
              <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
```

```
84      </runnables>
        <runnables name="Runnable_State_1" callback="false" service="false">
86        <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
88            <value xsi:type="am:NeedDeviation">
                <deviation>
90                <lowerBound xsi:type="am:LongObject" value="59400" />
                  <upperBound xsi:type="am:LongObject" value="60000" />
92                <distribution xsi:type="am:UniformDistribution" />
                </deviation>
94            </value>
            </default>
96        </runnableItems>
        </runnables>
98      <runnables name="Runnable_State_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
100         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
102             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700000" />
104               <upperBound xsi:type="am:LongObject" value="30000000" />
                  <distribution xsi:type="am:UniformDistribution" />
106             </deviation>
              </value>
108         </default>
          </runnableItems>
110     </runnables>
        <runnables name="Runnable_1" callback="false" service="false">
112       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
114           <value xsi:type="am:NeedDeviation">
                <deviation>
116               <lowerBound xsi:type="am:LongObject" value="5940000" />
                  <upperBound xsi:type="am:LongObject" value="6000000" />
118               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
120           </value>
            </default>
122       </runnableItems>
        </runnables>
124     <runnables name="Runnable_1_0" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/MessageContent_0?type=ModeLiteral" />
126     </runnables>
        <runnables name="Runnable_Transition_0" callback="false" service="false">
128       <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_0?type=ModeLiteral" />
        </runnables>
130     <runnables name="Runnable_Transition_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_1?type=ModeLiteral" />
132     </runnables>
        <runnables name="Runnable_Transition_2" callback="false" service="false">
134       <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_2?type=ModeLiteral" />
        </runnables>
136     <modes name="State">
          <literals name="State_0">
```

```
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="State_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="State_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
    <modes name="Message">
      <literals name="MessageContent_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageContent_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
```

```
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="100" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_Task_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.19: Variation 4 of State Machine.

A.1.3.5. Variation 5

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
          <entries probability="25.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_Task_2?type=
              InterProcessStimulus" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="State_1">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_1?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
          <entries name="State_0">
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_0?type=ModeLiteral"/>
            </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch" />
          </entries>
          <entries name="State_2">
```

```
52            <condition>
                <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_2?type=ModeLiteral"/>
54            </condition>
              <items xsi:type="am:CallSequence" name="CallSequence_State_2">
56              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
              </items>
58            <items xsi:type="am:ModeSwitch" />
            </entries>
60        </graphEntries>
      </callGraph>
62    <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
64    </customProperties>
      <customProperties key="osekTaskGroup">
66      <value xsi:type="am:StringObject" value="2" />
      </customProperties>
68  </tasks>
    <runnables name="Runnable_1_1" callback="false" service="false">
70    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_1?type=ModeLiteral" />
    </runnables>
72  <runnables name="Runnable_State_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
74      <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
76          <deviation>
              <lowerBound xsi:type="am:LongObject" value="59" />
78            <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
80          </deviation>
          </value>
82      </default>
      </runnableItems>
84  </runnables>
    <runnables name="Runnable_State_1" callback="false" service="false">
86    <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
88        <value xsi:type="am:NeedDeviation">
            <deviation>
90            <lowerBound xsi:type="am:LongObject" value="59400" />
              <upperBound xsi:type="am:LongObject" value="60000" />
92            <distribution xsi:type="am:UniformDistribution" />
            </deviation>
94        </value>
        </default>
96    </runnableItems>
    </runnables>
98  <runnables name="Runnable_State_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
100     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
102         <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
104           <upperBound xsi:type="am:LongObject" value="30000000" />
              <distribution xsi:type="am:UniformDistribution" />
106         </deviation>
          </value>
```

```
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
    <modes name="State">
      <literals name="State_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="State_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="State_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
    <modes name="Message">
      <literals name="MessageContent_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageContent_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
```

```
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_Task_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.20: Variation 5 of State Machine.

A.1.3.6. Variation 6

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
          </entries>
          <entries probability="25.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
```

```
20          <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_Task_2?type=
                InterProcessStimulus" />
          </graphEntries>
22      </callGraph>
        <customProperties key="priority">
24        <value xsi:type="am:StringObject" value="1" />
        </customProperties>
26      <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="1" />
28      </customProperties>
      </tasks>
30    <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
        <callGraph>
32        <graphEntries xsi:type="am:ModeSwitch">
          <entries name="State_1">
34          <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_1?type=ModeLiteral"/>
36          </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_1">
38            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
            </items>
40          <items xsi:type="am:ModeSwitch" />
          </entries>
42        <entries name="State_0">
            <condition>
44            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_0?type=ModeLiteral"/>
            </condition>
46          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
48          </items>
            <items xsi:type="am:ModeSwitch" />
50        </entries>
          <entries name="State_2">
52          <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_2?type=ModeLiteral"/>
54          </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_2">
56            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
            </items>
58          <items xsi:type="am:ModeSwitch" />
          </entries>
60      </graphEntries>
        </callGraph>
62      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="2" />
64      </customProperties>
        <customProperties key="osekTaskGroup">
66        <value xsi:type="am:StringObject" value="2" />
        </customProperties>
68    </tasks>
      <runnables name="Runnable_1_1" callback="false" service="false">
70      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/MessageContent_1?type=ModeLiteral" />
      </runnables>
72    <runnables name="Runnable_State_0" callback="false" service="false">
```

```
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="58" />
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
            <upperBound xsi:type="am:LongObject" value="60600" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700000" />
            <upperBound xsi:type="am:LongObject" value="30300000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/MessageContent_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_0?type=ModeLiteral" />
```

```
    </runnables>
    <runnables name="Runnable_Transition_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
    <modes name="State">
      <literals name="State_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="State_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="State_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
    <modes name="Message">
      <literals name="MessageContent_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageContent_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
```

```
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_Task_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
```

```
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
230 <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
232 <componentsModel />
</am:Amalthea>
```

Listing A.21: Variation 6 of State Machine.

A.1.3.7. Variation 7

```
<?xml version="1.0" encoding="UTF-8"?>
2 <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
4   <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
6       <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="75.0">
8           <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
10          </items>
          </entries>
12        <entries probability="25.0">
            <items xsi:type="am:CallSequence" name="CallSequence_1_1">
14            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
            </items>
16        </entries>
        </graphEntries>
18      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1_2">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
20        <calls xsi:type="am:InterProcessTrigger" stimulus="Stimulus_Task_2?type=
              InterProcessStimulus" />
        </graphEntries>
22    </callGraph>
      <customProperties key="priority">
24      <value xsi:type="am:StringObject" value="1" />
      </customProperties>
26    <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
28    </customProperties>
    </tasks>
30  <tasks name="Task_2" preemption="preemptive" multipleTaskActivationLimit="1">
      <callGraph>
32      <graphEntries xsi:type="am:ModeSwitch">
          <entries name="State_1">
34          <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_1?type=ModeLiteral"/>
36          </condition>
            <items xsi:type="am:CallSequence" name="CallSequence_State_1">
38            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
            </items>
40          <items xsi:type="am:ModeSwitch" />
```

```
        </entries>
        <entries name="State_0">
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_0?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch" />
        </entries>
        <entries name="State_2">
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_2?type=ModeLiteral"/>
          </condition>
          <items xsi:type="am:CallSequence" name="CallSequence_State_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch" />
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/MessageContent_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_State_0" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="58" />
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
            <upperBound xsi:type="am:LongObject" value="60600" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
```

```
96      </runnableItems>
    </runnables>
98  <runnables name="Runnable_State_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
100     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
102         <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
104           <upperBound xsi:type="am:LongObject" value="30300000" />
              <distribution xsi:type="am:UniformDistribution" />
106         </deviation>
          </value>
108     </default>
      </runnableItems>
110 </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
112   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
114       <value xsi:type="am:NeedDeviation">
            <deviation>
116           <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
118           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
120       </value>
        </default>
122   </runnableItems>
    </runnables>
124 <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/MessageContent_0?type=ModeLiteral" />
126 </runnables>
    <runnables name="Runnable_Transition_0" callback="false" service="false">
128   <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
    </runnables>
130 <runnables name="Runnable_Transition_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
132 </runnables>
    <runnables name="Runnable_Transition_2" callback="false" service="false">
134   <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
136 <modes name="State">
      <literals name="State_0">
138     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
140     </customProperties>
      </literals>
142   <literals name="State_1">
        <customProperties key="enumValue">
144       <value xsi:type="am:LongObject" value="1" />
        </customProperties>
146   </literals>
      <literals name="State_2">
148     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
```

```
150         </customProperties>
          </literals>
152       </modes>
          <modes name="Message">
154         <literals name="MessageContent_0">
            <customProperties key="enumValue">
156           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
158         </literals>
            <literals name="MessageContent_1">
160         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
162         </customProperties>
            </literals>
164       </modes>
          <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
166         <size value="8" unit="bit" />
          </modeLabels>
168       <modeLabels name="message" initialValue="Message/MessageContent_0?type=ModeLiteral">
            <size value="1" unit="bit" />
170       </modeLabels>
        </swModel>
172     <hwModel>
          <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
              IPC_1.0?type=HwFeature" puType="CPU"/>
174       <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
          </definitions>
176       <featureCategories name="Instructions" featureType="performance">
            <features name="IPC_1.0" value="1.0" />
178       </featureCategories>
          <structures name="System" structureType="System">
180         <structures name="Ecu_1" structureType="ECU">
              <structures name="Processor_1" structureType="Microcontroller">
182             <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                    FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
                </modules>
184             <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                    FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
                  <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
186             </modules>
              </structures>
188         </structures>
          </structures>
190       <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
            <defaultValue value="600.0" unit="MHz"/>
192       </domains>
        </hwModel>
194     <osModel>
          <operatingSystems name="Generic_OS">
196         <taskSchedulers name="Scheduler_1">
              <schedulingAlgorithm xsi:type="am:OSEK" />
198         </taskSchedulers>
            <osDataConsistency mode="noProtection" />
200       </operatingSystems>
        </osModel>
202     <stimuliModel>
          <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
204         <offset value="0" unit="ms" />
```

```
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="Stimulus_Task_2" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="state?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.22: Variation 7 of State Machine.

A.1.4. Feedback Loop

A.1.4.1. Variation 1

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
```

```
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CS_e_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_e_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_R1">
        <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
```

```
58              </condition>
              </entries>
60            </items>
              <condition>
62              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_0?type=ModeLiteral" />
              </condition>
64          </entries>
            <entries>
66            <items xsi:type="am:CallSequence" name="CS_state_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
68              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
              </items>
70            <items xsi:type="am:ModeSwitch">
                <entries>
72                <items xsi:type="am:CallSequence" name="CS_1_to_0">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
74                </items>
                  <condition>
76                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
                  </condition>
78              </entries>
                <entries>
80                <items xsi:type="am:CallSequence" name="CS_1_to_2">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
82                </items>
                  <condition>
84                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
                  </condition>
86              </entries>
              </items>
88            <condition>
                <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_1?type=ModeLiteral" />
90            </condition>
            </entries>
92          <entries>
              <items xsi:type="am:CallSequence" name="CS_state_2">
94              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
96            </items>
              <items xsi:type="am:ModeSwitch">
98              <entries>
                  <items xsi:type="am:CallSequence" name="CS_2_to_1">
100                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                  </items>
102               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
104                 </condition>
                </entries>
106             <entries>
                  <items xsi:type="am:CallSequence" name="CS_2_to_2">
108                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                  </items>
110               <condition>
```

```
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
112           </condition>
            </entries>
114         </items>
            <condition>
116           <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_2?type=ModeLiteral" />
            </condition>
118       </entries>
      </graphEntries>
120   <graphEntries xsi:type="am:ProbabilitySwitch">
        <entries probability="0.3">
122       <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
            <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                InterProcessStimulus" />
124       </items>
        </entries>
126     <entries probability="0.7">
          <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
128     </entries>
      </graphEntries>
130   <graphEntries xsi:type="am:CallSequence" name="CS_R2">
        <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
132   </graphEntries>
    </callGraph>
134 <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
136 </customProperties>
    <customProperties key="osekTaskGroup">
138   <value xsi:type="am:StringObject" value="2" />
    </customProperties>
140 </tasks>
    <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
142   <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
144       <entries>
            <items xsi:type="am:CallSequence" name="CS_y_0">
146           <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
            </items>
148         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                  ModeLiteral" />
150         </condition>
          </entries>
152       <entries>
            <items xsi:type="am:CallSequence" name="CS_y_1">
154           <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
            </items>
156         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                  =ModeLiteral" />
158         </condition>
          </entries>
160       <entries>
            <items xsi:type="am:CallSequence" name="CS_y_2">
162           <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
```

```
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.3">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.7">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.5">
              <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.5">
              <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                =ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.7">
```

```
              <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
218             <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
220         </entries>
            <entries probability="0.3">
222           <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
224           </items>
            </entries>
226       </items>
          <condition>
228         <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                type=ModeLiteral" />
          </condition>
230     </entries>
      </graphEntries>
232   <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
        <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
234   </graphEntries>
    </callGraph>
236 <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
238 </customProperties>
    <customProperties key="osekTaskGroup">
240   <value xsi:type="am:StringObject" value="1" />
    </customProperties>
242 </tasks>
  <runnables name="Set_e_0" callback="false" service="false">
244   <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_0?type=ModeLiteral" />
  </runnables>
246 <runnables name="Set_e_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_1?type=ModeLiteral" />
248 </runnables>
  <runnables name="Set_state_0" callback="false" service="false">
250   <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
  </runnables>
252 <runnables name="Set_state_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
254 </runnables>
  <runnables name="Set_state_2" callback="false" service="false">
256   <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
  </runnables>
258 <runnables name="Set_y_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_0?type=ModeLiteral" />
260 </runnables>
  <runnables name="Set_y_50" callback="false" service="false">
262   <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_50?type=ModeLiteral" />
  </runnables>
264 <runnables name="Set_y_100" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_100?type=ModeLiteral" />
```

```
    </runnables>
    <runnables name="R_3_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400000" />
              <upperBound xsi:type="am:LongObject" value="60000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Set_w_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_50?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_100" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_100?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_u_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
          modeValue="U/U_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_u_1" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
          modeValue="U/U_1?type=ModeLiteral" />
    </runnables>
    <runnables name="R1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <modes name="E">
      <literals name="E_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="E_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="U">
      <literals name="U_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
```

```
376         </customProperties>
          </literals>
378       <literals name="U_1">
            <customProperties key="enumValue">
380           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
382       </literals>
        </modes>
384     <modes name="Y">
          <literals name="Y_0">
386         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
388         </customProperties>
          </literals>
390       <literals name="Y_50">
            <customProperties key="enumValue">
392           <value xsi:type="am:LongObject" value="50" />
            </customProperties>
394       </literals>
          <literals name="Y_100">
396         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="100" />
398         </customProperties>
          </literals>
400     </modes>
        <modes name="W">
402       <literals name="W_0">
            <customProperties key="enumValue">
404           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
406       </literals>
          <literals name="W_50">
408         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="50" />
410         </customProperties>
          </literals>
412       <literals name="W_100">
            <customProperties key="enumValue">
414           <value xsi:type="am:LongObject" value="100" />
            </customProperties>
416       </literals>
        </modes>
418     <modes name="State">
          <literals name="State_0">
420         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
422         </customProperties>
          </literals>
424       <literals name="State_1">
            <customProperties key="enumValue">
426           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
428       </literals>
          <literals name="State_2">
430         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
432         </customProperties>
          </literals>
```

```
434     </modes>
        <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
436       <size value="1" unit="bit" />
        </modeLabels>
438     <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
          <size value="8" unit="bit" />
440     </modeLabels>
        <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
442       <size value="8" unit="bit" />
        </modeLabels>
444     <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
          <size value="1" unit="bit" />
446     </modeLabels>
        <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
448       <size value="8" unit="bit" />
        </modeLabels>
450   </swModel>
      <hwModel>
452     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
454     </definitions>
        <featureCategories name="Instructions" featureType="performance">
456       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
458     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
460         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
462           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
464             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
466         </structures>
          </structures>
468     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
470       <defaultValue value="600.0" unit="MHz"/>
        </domains>
472   </hwModel>
      <osModel>
474     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
476         <schedulingAlgorithm xsi:type="am:OSEK" />
          </taskSchedulers>
478       <osDataConsistency mode="noProtection" />
        </operatingSystems>
480   </osModel>
      <stimuliModel>
482     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
          <offset value="0" unit="ms" />
484       <recurrence value="600" unit="ms" />
        </stimuli>
486     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
          <offset value="20" unit="ms" />
488       <recurrence value="300" unit="ms" />
```

```
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="50" unit="ms" />
      <recurrence value="500" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
        =ModeLabel" />
```

```
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
        type=ModeLabel" />
536 <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
        ?type=ModeLabel" />
  </mappingModel>
538 <componentsModel />
</am:Amalthea>
```

Listing A.23: Variation 1 of Feedback Loop.

A.1.4.2. Variation 2

```
<?xml version="1.0" encoding="UTF-8"?>
2 <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
4   <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
6       <graphEntries xsi:type="am:ModeSwitch">
          <entries>
8           <items xsi:type="am:CallSequence" name="CS_e_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
10          </items>
            <condition>
12            <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                  ModeLiteral" />
            </condition>
14        </entries>
          <entries>
16          <items xsi:type="am:CallSequence" name="CS_e_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
18          </items>
            <condition>
20            <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                  ModeLiteral" />
            </condition>
22        </entries>
        </graphEntries>
24      <graphEntries xsi:type="am:CallSequence" name="CS_R1">
          <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
26      </graphEntries>
      </callGraph>
28    <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
30    </customProperties>
      <customProperties key="osekTaskGroup">
32      <value xsi:type="am:StringObject" value="3" />
      </customProperties>
34  </tasks>
    <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
36    <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
38        <entries>
            <items xsi:type="am:CallSequence" name="CS_state_0">
```

```
40              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
42            </items>
              <items xsi:type="am:ModeSwitch">
44              <entries>
                  <items xsi:type="am:CallSequence" name="CS_0_to_0">
46                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
                  </items>
48                <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
50                </condition>
                </entries>
52              <entries>
                  <items xsi:type="am:CallSequence" name="CS_0_to_1">
54                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                  </items>
56                <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
58                </condition>
                </entries>
60            </items>
              <condition>
62              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_0?type=ModeLiteral" />
              </condition>
64          </entries>
            <entries>
66            <items xsi:type="am:CallSequence" name="CS_state_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
68              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
              </items>
70            <items xsi:type="am:ModeSwitch">
                <entries>
72                <items xsi:type="am:CallSequence" name="CS_1_to_0">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
74                </items>
                  <condition>
76                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
                  </condition>
78              </entries>
                <entries>
80                <items xsi:type="am:CallSequence" name="CS_1_to_2">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
82                </items>
                  <condition>
84                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
                  </condition>
86              </entries>
              </items>
88            <condition>
                <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_1?type=ModeLiteral" />
90            </condition>
            </entries>
```

```
92          <entries>
              <items xsi:type="am:CallSequence" name="CS_state_2">
94              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
96            </items>
              <items xsi:type="am:ModeSwitch">
98              <entries>
                  <items xsi:type="am:CallSequence" name="CS_2_to_1">
100                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                  </items>
102               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
104               </condition>
                </entries>
106             <entries>
                  <items xsi:type="am:CallSequence" name="CS_2_to_2">
108                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                  </items>
110               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
112               </condition>
                </entries>
114           </items>
              <condition>
116             <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_2?type=ModeLiteral" />
              </condition>
118         </entries>
          </graphEntries>
120       <graphEntries xsi:type="am:ProbabilitySwitch">
            <entries probability="0.3">
122           <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
                <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                    InterProcessStimulus" />
124           </items>
            </entries>
126         <entries probability="0.7">
              <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
128         </entries>
          </graphEntries>
130       <graphEntries xsi:type="am:CallSequence" name="CS_R2">
            <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
132       </graphEntries>
        </callGraph>
134     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="2" />
136     </customProperties>
        <customProperties key="osekTaskGroup">
138       <value xsi:type="am:StringObject" value="2" />
        </customProperties>
140   </tasks>
      <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
142     <callGraph>
          <graphEntries xsi:type="am:ModeSwitch">
144         <entries>
```

```
          <items xsi:type="am:CallSequence" name="CS_y_0">
146         <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
          </items>
148       <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                ModeLiteral" />
150       </condition>
        </entries>
152     <entries>
          <items xsi:type="am:CallSequence" name="CS_y_1">
154         <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
          </items>
156       <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                =ModeLiteral" />
158       </condition>
        </entries>
160     <entries>
          <items xsi:type="am:CallSequence" name="CS_y_2">
162         <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
          </items>
164       <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                type=ModeLiteral" />
166       </condition>
        </entries>
168   </graphEntries>
    </callGraph>
170 <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
172 </customProperties>
    <customProperties key="osekTaskGroup">
174   <value xsi:type="am:StringObject" value="1" />
    </customProperties>
176 </tasks>
    <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
178 <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
180     <entries>
          <items xsi:type="am:ProbabilitySwitch">
182         <entries probability="0.3">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
184             <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
186         </entries>
            <entries probability="0.7">
188           <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
190           </items>
            </entries>
192       </items>
          <condition>
194         <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                ModeLiteral" />
          </condition>
196     </entries>
        <entries>
```

```
198             <items xsi:type="am:ProbabilitySwitch">
                  <entries probability="0.5">
200                 <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
202                 </items>
                  </entries>
204               <entries probability="0.5">
                    <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
206                   <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
                    </items>
208               </entries>
                </items>
210             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                      =ModeLiteral" />
212             </condition>
              </entries>
214           <entries>
                <items xsi:type="am:ProbabilitySwitch">
216               <entries probability="0.7">
                    <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
218                   <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                    </items>
220               </entries>
                  <entries probability="0.3">
222                 <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
224                 </items>
                  </entries>
226             </items>
                <condition>
228               <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                      type=ModeLiteral" />
                </condition>
230           </entries>
          </graphEntries>
232       <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
234       </graphEntries>
        </callGraph>
236     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
238     </customProperties>
        <customProperties key="osekTaskGroup">
240       <value xsi:type="am:StringObject" value="1" />
        </customProperties>
242   </tasks>
      <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
244     <callGraph>
          <graphEntries xsi:type="am:ProbabilitySwitch">
246         <entries probability="15.0">
              <items xsi:type="am:CallSequence" name="CallSequence_5_0">
248             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_0?type=Runnable" />
              </items>
250         </entries>
            <entries probability="20.0">
252           <items xsi:type="am:CallSequence" name="CallSequence_5_1">
```

```
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
          </items>
        </entries>
        <entries probability="30.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_4?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_3?type=Runnable" />
          </items>
          <condition>
```

```
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_3?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_4?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_4?type=ModeLiteral" />
          </condition>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_6_x">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_x?type=Runnable" />
          </items>
        </defaultEntry>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <runnables name="Set_e_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
        modeValue="E/E_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_e_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
        modeValue="E/E_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_state_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_state_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_state_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
        modeValue="State/State_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_y_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
        modeValue="Y/Y_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_y_50" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
        modeValue="Y/Y_50?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_y_100" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
        modeValue="Y/Y_100?type=ModeLiteral" />
```

```
    </runnables>
    <runnables name="R_3_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400000" />
              <upperBound xsi:type="am:LongObject" value="60000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Set_w_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_50?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_100" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_100?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_u_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
          modeValue="U/U_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_u_1" callback="false" service="false">
```

```
408       <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
              modeValue="U/U_1?type=ModeLiteral" />
          </runnables>
410       <runnables name="R1" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
412           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
414               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="5940000" />
416                 <upperBound xsi:type="am:LongObject" value="6000000" />
                    <distribution xsi:type="am:UniformDistribution" />
418               </deviation>
                </value>
420           </default>
            </runnableItems>
422       </runnables>
          <runnables name="R2" callback="false" service="false">
424         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
426             <value xsi:type="am:NeedDeviation">
                  <deviation>
428                 <lowerBound xsi:type="am:LongObject" value="594000" />
                    <upperBound xsi:type="am:LongObject" value="600000" />
430                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
432             </value>
              </default>
434         </runnableItems>
          </runnables>
436       <runnables name="R_4" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
438           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
440               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="5940000" />
442                 <upperBound xsi:type="am:LongObject" value="6000000" />
                    <distribution xsi:type="am:UniformDistribution" />
444               </deviation>
                </value>
446           </default>
            </runnableItems>
448       </runnables>
          <runnables name="Runnable_5" callback="false" service="false">
450         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
452             <value xsi:type="am:NeedDeviation">
                  <deviation>
454                 <lowerBound xsi:type="am:LongObject" value="5940000" />
                    <upperBound xsi:type="am:LongObject" value="6000000" />
456                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
458             </value>
              </default>
460         </runnableItems>
          </runnables>
462       <runnables name="Runnable_5_0" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_0?type=ModeLiteral" />
```

```
464     </runnables>
        <runnables name="Runnable_5_1" callback="false" service="false">
466       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_1?type=ModeLiteral" />
        </runnables>
468     <runnables name="Runnable_5_2" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_2?type=ModeLiteral" />
470     </runnables>
        <runnables name="Runnable_5_3" callback="false" service="false">
472       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_3?type=ModeLiteral" />
        </runnables>
474     <runnables name="Runnable_5_4" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_4?type=ModeLiteral" />
476     </runnables>
        <runnables name="Runnable_6_x" callback="false" service="false">
478       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
480           <value xsi:type="am:NeedDeviation">
                <deviation>
482               <lowerBound xsi:type="am:LongObject" value="29700000" />
                  <upperBound xsi:type="am:LongObject" value="30000000" />
484               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
486           </value>
            </default>
488       </runnableItems>
        </runnables>
490     <runnables name="Runnable_6_1" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
492         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
494             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="5940000" />
496               <upperBound xsi:type="am:LongObject" value="6000000" />
                  <distribution xsi:type="am:UniformDistribution" />
498             </deviation>
              </value>
500         </default>
          </runnableItems>
502     </runnables>
        <runnables name="Runnable_6_2" callback="false" service="false">
504       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
506           <value xsi:type="am:NeedDeviation">
                <deviation>
508               <lowerBound xsi:type="am:LongObject" value="594" />
                  <upperBound xsi:type="am:LongObject" value="600" />
510               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
512           </value>
            </default>
514       </runnableItems>
        </runnables>
516     <runnables name="Runnable_6_3" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
```

```
518         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
520             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700" />
522               <upperBound xsi:type="am:LongObject" value="30000" />
                  <distribution xsi:type="am:UniformDistribution" />
524             </deviation>
              </value>
526         </default>
        </runnableItems>
528   </runnables>
      <runnables name="Runnable_6_4" callback="false" service="false">
530     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
532         <value xsi:type="am:NeedDeviation">
              <deviation>
534             <lowerBound xsi:type="am:LongObject" value="594000" />
                <upperBound xsi:type="am:LongObject" value="600000" />
536             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
538         </value>
          </default>
540     </runnableItems>
      </runnables>
542   <modes name="E">
        <literals name="E_0">
544       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
546       </customProperties>
        </literals>
548     <literals name="E_1">
          <customProperties key="enumValue">
550         <value xsi:type="am:LongObject" value="1" />
          </customProperties>
552     </literals>
      </modes>
554   <modes name="U">
        <literals name="U_0">
556       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
558       </customProperties>
        </literals>
560     <literals name="U_1">
          <customProperties key="enumValue">
562         <value xsi:type="am:LongObject" value="1" />
          </customProperties>
564     </literals>
      </modes>
566   <modes name="Y">
        <literals name="Y_0">
568       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
570       </customProperties>
        </literals>
572     <literals name="Y_50">
          <customProperties key="enumValue">
574         <value xsi:type="am:LongObject" value="50" />
          </customProperties>
```

```
576       </literals>
          <literals name="Y_100">
578         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="100" />
580         </customProperties>
          </literals>
582     </modes>
        <modes name="W">
584       <literals name="W_0">
            <customProperties key="enumValue">
586           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
588       </literals>
          <literals name="W_50">
590         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="50" />
592         </customProperties>
          </literals>
594       <literals name="W_100">
            <customProperties key="enumValue">
596           <value xsi:type="am:LongObject" value="100" />
            </customProperties>
598       </literals>
        </modes>
600     <modes name="State">
          <literals name="State_0">
602         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
604         </customProperties>
          </literals>
606       <literals name="State_1">
            <customProperties key="enumValue">
608           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
610       </literals>
          <literals name="State_2">
612         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
614         </customProperties>
          </literals>
616     </modes>
        <modes name="Message">
618       <literals name="Message_0">
            <customProperties key="enumValue">
620           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
622       </literals>
          <literals name="Message_1">
624         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
626         </customProperties>
          </literals>
628       <literals name="Message_2">
            <customProperties key="enumValue">
630           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
632       </literals>
          <literals name="Message_3">
```

```
634         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="3" />
636         </customProperties>
          </literals>
638       <literals name="Message_4">
            <customProperties key="enumValue">
640           <value xsi:type="am:LongObject" value="4" />
            </customProperties>
642       </literals>
        </modes>
644     <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
          <size value="1" unit="bit" />
646     </modeLabels>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
648       <size value="8" unit="bit" />
        </modeLabels>
650     <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
          <size value="8" unit="bit" />
652     </modeLabels>
        <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
654       <size value="8" unit="bit" />
        </modeLabels>
656     <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
          <size value="1" unit="bit" />
658     </modeLabels>
        <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
660       <size value="8" unit="bit" />
        </modeLabels>
662   </swModel>
      <hwModel>
664     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
666     </definitions>
        <featureCategories name="Instructions" featureType="performance">
668       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
670     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
672         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
674           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
676             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
678         </structures>
          </structures>
680     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
682       <defaultValue value="600.0" unit="MHz"/>
        </domains>
684   </hwModel>
      <osModel>
686     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
688         <schedulingAlgorithm xsi:type="am:OSEK" />
```

```
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="600" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="20" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="50" unit="ms" />
      <recurrence value="500" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="100" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
      <offset value="15" unit="ms" />
      <recurrence value="60" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
```

```
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" entity="Stimulus_Task_6?type=
        PeriodicStimulus" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_0" entity="Runnable_5_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_3" entity="Runnable_5_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_4" entity="Runnable_5_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_1" entity="Runnable_6_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_2" entity="Runnable_6_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_3" entity="Runnable_6_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_4" entity="Runnable_6_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_x" entity="Runnable_6_x?type=
        Runnable" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
        ?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="26" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
```

```
</am:Amalthea>
```

Listing A.24: Variation 2 of Feedback Loop.

A.1.4.3. Variation 3

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_e_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                  ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_e_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                  ModeLiteral" />
            </condition>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_R1">
          <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_state_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CS_0_to_0">
```

```
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
```

```
98              <entries>
                  <items xsi:type="am:CallSequence" name="CS_2_to_1">
100                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                  </items>
102               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
104               </condition>
                </entries>
106             <entries>
                  <items xsi:type="am:CallSequence" name="CS_2_to_2">
108                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                  </items>
110               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
112               </condition>
                </entries>
114           </items>
              <condition>
116             <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_2?type=ModeLiteral" />
              </condition>
118         </entries>
          </graphEntries>
120       <graphEntries xsi:type="am:ProbabilitySwitch">
            <entries probability="0.3">
122           <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
                <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                    InterProcessStimulus" />
124           </items>
            </entries>
126         <entries probability="0.7">
              <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
128         </entries>
          </graphEntries>
130       <graphEntries xsi:type="am:CallSequence" name="CS_R2">
            <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
132       </graphEntries>
        </callGraph>
134     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="2" />
136     </customProperties>
        <customProperties key="osekTaskGroup">
138       <value xsi:type="am:StringObject" value="2" />
        </customProperties>
140   </tasks>
      <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
142     <callGraph>
          <graphEntries xsi:type="am:ModeSwitch">
144         <entries>
              <items xsi:type="am:CallSequence" name="CS_y_0">
146             <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
              </items>
148           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                    ModeLiteral" />
```

```
150           </condition>
            </entries>
152           <entries>
              <items xsi:type="am:CallSequence" name="CS_y_1">
154             <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
              </items>
156           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                    =ModeLiteral" />
158           </condition>
            </entries>
160           <entries>
              <items xsi:type="am:CallSequence" name="CS_y_2">
162             <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
              </items>
164           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                    type=ModeLiteral" />
166           </condition>
            </entries>
168       </graphEntries>
        </callGraph>
170     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
172     </customProperties>
        <customProperties key="osekTaskGroup">
174       <value xsi:type="am:StringObject" value="1" />
        </customProperties>
176   </tasks>
      <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
178     <callGraph>
          <graphEntries xsi:type="am:ModeSwitch">
180         <entries>
              <items xsi:type="am:ProbabilitySwitch">
182             <entries probability="0.3">
                  <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
184                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                  </items>
186             </entries>
                <entries probability="0.7">
188               <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
190               </items>
                </entries>
192           </items>
              <condition>
194             <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                    ModeLiteral" />
              </condition>
196         </entries>
            <entries>
198           <items xsi:type="am:ProbabilitySwitch">
                <entries probability="0.5">
200               <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
202               </items>
                </entries>
```

```
204             <entries probability="0.5">
                  <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
206                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
                  </items>
208             </entries>
              </items>
210           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                    =ModeLiteral" />
212           </condition>
            </entries>
214         <entries>
              <items xsi:type="am:ProbabilitySwitch">
216             <entries probability="0.7">
                  <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
218                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                  </items>
220             </entries>
                <entries probability="0.3">
222               <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
224               </items>
                </entries>
226           </items>
              <condition>
228             <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                    type=ModeLiteral" />
              </condition>
230         </entries>
          </graphEntries>
232       <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
234       </graphEntries>
        </callGraph>
236     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
238     </customProperties>
        <customProperties key="osekTaskGroup">
240       <value xsi:type="am:StringObject" value="1" />
        </customProperties>
242   </tasks>
      <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
244     <callGraph>
          <graphEntries xsi:type="am:ProbabilitySwitch">
246         <entries probability="15.0">
              <items xsi:type="am:CallSequence" name="CallSequence_5_0">
248             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_0?type=Runnable" />
              </items>
250         </entries>
            <entries probability="20.0">
252           <items xsi:type="am:CallSequence" name="CallSequence_5_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
254           </items>
            </entries>
256         <entries probability="30.0">
              <items xsi:type="am:CallSequence" name="CallSequence_5_2">
258             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
```

```
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_4?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="Stimulus_Task_6?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_3?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_3?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_4?type=Runnable" />
```

```
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_4?type=ModeLiteral" />
            </condition>
          </entries>
          <defaultEntry>
            <items xsi:type="am:CallSequence" name="CallSequence_6_x">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_x?type=Runnable" />
            </items>
          </defaultEntry>
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
    </tasks>
    <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CS_Task_7">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
        </graphEntries>
      </callGraph>
    </tasks>
    <runnables name="Set_e_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_e_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_state_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_state_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_state_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_50?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_100" callback="false" service="false">
```

```
362       <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
              modeValue="Y/Y_100?type=ModeLiteral" />
          </runnables>
364       <runnables name="R_3_0" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
366           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
368               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="594000" />
370                 <upperBound xsi:type="am:LongObject" value="600000" />
                    <distribution xsi:type="am:UniformDistribution" />
372               </deviation>
                </value>
374           </default>
            </runnableItems>
376       </runnables>
          <runnables name="R_3_2" callback="false" service="false">
378         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
380             <value xsi:type="am:NeedDeviation">
                  <deviation>
382                 <lowerBound xsi:type="am:LongObject" value="59400000" />
                    <upperBound xsi:type="am:LongObject" value="60000000" />
384                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
386             </value>
              </default>
388         </runnableItems>
          </runnables>
390       <runnables name="R_3_1" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
392           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
394               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="5940000" />
396                 <upperBound xsi:type="am:LongObject" value="6000000" />
                    <distribution xsi:type="am:UniformDistribution" />
398               </deviation>
                </value>
400           </default>
            </runnableItems>
402       </runnables>
          <runnables name="Set_w_0" callback="false" service="false">
404         <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_0?type=ModeLiteral" />
          </runnables>
406       <runnables name="Set_w_50" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_50?type=ModeLiteral" />
408       </runnables>
          <runnables name="Set_w_100" callback="false" service="false">
410         <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_100?type=ModeLiteral" />
          </runnables>
412       <runnables name="Set_u_0" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
              modeValue="U/U_0?type=ModeLiteral" />
414       </runnables>
```

```
    <runnables name="Set_u_1" callback="false" service="false">
416   <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
          modeValue="U/U_1?type=ModeLiteral" />
    </runnables>
418 <runnables name="R1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
420     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
422         <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
424           <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
426         </deviation>
          </value>
428     </default>
      </runnableItems>
430 </runnables>
    <runnables name="R2" callback="false" service="false">
432   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
434       <value xsi:type="am:NeedDeviation">
            <deviation>
436           <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
438           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
440       </value>
        </default>
442   </runnableItems>
    </runnables>
444 <runnables name="R_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
446     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
448         <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
450           <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
452         </deviation>
          </value>
454     </default>
      </runnableItems>
456 </runnables>
    <runnables name="Runnable_5" callback="false" service="false">
458   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
460       <value xsi:type="am:NeedDeviation">
            <deviation>
462           <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
464           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
466       </value>
        </default>
468   </runnableItems>
    </runnables>
470 <runnables name="Runnable_5_0" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_5_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_5_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_5_3" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_3?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_5_4" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_4?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_6_x" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
              <upperBound xsi:type="am:LongObject" value="30000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="600" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
```

```
524     <runnables name="Runnable_6_3" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
526         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
528             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700" />
530               <upperBound xsi:type="am:LongObject" value="30000" />
                  <distribution xsi:type="am:UniformDistribution" />
532             </deviation>
              </value>
534         </default>
          </runnableItems>
536     </runnables>
        <runnables name="Runnable_6_4" callback="false" service="false">
538       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
540           <value xsi:type="am:NeedDeviation">
                <deviation>
542               <lowerBound xsi:type="am:LongObject" value="594000" />
                  <upperBound xsi:type="am:LongObject" value="600000" />
544               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
546           </value>
            </default>
548       </runnableItems>
        </runnables>
550     <runnables name="Runnable_7_1" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
552         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
554             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="35640000" />
556               <upperBound xsi:type="am:LongObject" value="36000000" />
                  <distribution xsi:type="am:UniformDistribution" />
558             </deviation>
              </value>
560         </default>
          </runnableItems>
562     </runnables>
        <runnables name="Runnable_7_2" callback="false" service="false">
564       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
566           <value xsi:type="am:NeedDeviation">
                <deviation>
568               <lowerBound xsi:type="am:LongObject" value="11880000" />
                  <upperBound xsi:type="am:LongObject" value="12000000" />
570               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
572           </value>
            </default>
574       </runnableItems>
        </runnables>
576     <modes name="E">
          <literals name="E_0">
578       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
580       </customProperties>
        </literals>
```

```
582       <literals name="E_1">
            <customProperties key="enumValue">
584           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
586       </literals>
        </modes>
588     <modes name="U">
          <literals name="U_0">
590         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
592         </customProperties>
          </literals>
594       <literals name="U_1">
            <customProperties key="enumValue">
596           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
598       </literals>
        </modes>
600     <modes name="Y">
          <literals name="Y_0">
602         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
604         </customProperties>
          </literals>
606       <literals name="Y_50">
            <customProperties key="enumValue">
608           <value xsi:type="am:LongObject" value="50" />
            </customProperties>
610       </literals>
          <literals name="Y_100">
612         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="100" />
614         </customProperties>
          </literals>
616     </modes>
        <modes name="W">
618       <literals name="W_0">
            <customProperties key="enumValue">
620           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
622       </literals>
          <literals name="W_50">
624         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="50" />
626         </customProperties>
          </literals>
628       <literals name="W_100">
            <customProperties key="enumValue">
630           <value xsi:type="am:LongObject" value="100" />
            </customProperties>
632       </literals>
        </modes>
634     <modes name="State">
          <literals name="State_0">
636         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
638         </customProperties>
          </literals>
```

```
640       <literals name="State_1">
            <customProperties key="enumValue">
642           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
644       </literals>
          <literals name="State_2">
646         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
648         </customProperties>
          </literals>
650     </modes>
        <modes name="Message">
652       <literals name="Message_0">
            <customProperties key="enumValue">
654           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
656       </literals>
          <literals name="Message_1">
658         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
660         </customProperties>
          </literals>
662       <literals name="Message_2">
            <customProperties key="enumValue">
664           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
666       </literals>
          <literals name="Message_3">
668         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="3" />
670         </customProperties>
          </literals>
672       <literals name="Message_4">
            <customProperties key="enumValue">
674           <value xsi:type="am:LongObject" value="4" />
            </customProperties>
676       </literals>
        </modes>
678     <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
          <size value="1" unit="bit" />
680     </modeLabels>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
682       <size value="8" unit="bit" />
        </modeLabels>
684     <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
          <size value="8" unit="bit" />
686     </modeLabels>
        <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
688       <size value="8" unit="bit" />
        </modeLabels>
690     <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
          <size value="1" unit="bit" />
692     </modeLabels>
        <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
694       <size value="8" unit="bit" />
        </modeLabels>
696   </swModel>
      <hwModel>
```

```
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="600" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="20" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="50" unit="ms" />
      <recurrence value="500" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="100" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_6">
      <offset value="15" unit="ms" />
      <recurrence value="60" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
      <recurrence value="1000" unit="ms" />
    </stimuli>
```

```
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_6" entity="Stimulus_Task_6?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
        PeriodicStimulus" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_0" entity="Runnable_5_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_3" entity="Runnable_5_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_4" entity="Runnable_5_4?type=
        Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_1" entity="Runnable_6_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_2" entity="Runnable_6_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_3" entity="Runnable_6_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_4" entity="Runnable_6_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_x" entity="Runnable_6_x?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
        Runnable" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
        =ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
        type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
        ?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="26" abstractElement="
        message?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.25: Variation 3 of Feedback Loop.

A.1.4.4. Variation 4

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_e_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
```

```
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                  ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_e_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                  ModeLiteral" />
            </condition>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_R1">
          <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
               />
          <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_state_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CS_0_to_0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                      type=ModeLiteral" />
                </condition>
              </entries>
              <entries>
                <items xsi:type="am:CallSequence" name="CS_0_to_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                      type=ModeLiteral" />
                </condition>
              </entries>
            </items>
```

```
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_0?type=ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_state_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CS_1_to_0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                      type=ModeLiteral" />
                </condition>
              </entries>
              <entries>
                <items xsi:type="am:CallSequence" name="CS_1_to_2">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                      type=ModeLiteral" />
                </condition>
              </entries>
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_1?type=ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_state_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CS_2_to_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                      type=ModeLiteral" />
                </condition>
              </entries>
              <entries>
                <items xsi:type="am:CallSequence" name="CS_2_to_2">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                      type=ModeLiteral" />
                </condition>
```

```
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_2?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_IPA_T3">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_3?type=InterProcessStimulus"
             />
      </graphEntries>
      <graphEntries xsi:type="am:ProbabilitySwitch">
        <entries probability="0.3">
          <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
            <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                InterProcessStimulus" />
          </items>
        </entries>
        <entries probability="0.7">
          <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_R2">
        <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_3" stimuli="IPA_Task_3?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                =ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_2">
```

```
166             <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
            </items>
168         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                  type=ModeLiteral" />
170         </condition>
          </entries>
172     </graphEntries>
      </callGraph>
174   <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
176   </customProperties>
      <customProperties key="osekTaskGroup">
178     <value xsi:type="am:StringObject" value="1" />
      </customProperties>
180 </tasks>
    <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
182   <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
184       <entries>
            <items xsi:type="am:ProbabilitySwitch">
186           <entries probability="0.3">
                <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
188               <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                </items>
190           </entries>
              <entries probability="0.7">
192             <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
194             </items>
              </entries>
196         </items>
            <condition>
198           <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                  ModeLiteral" />
            </condition>
200       </entries>
          <entries>
202         <items xsi:type="am:ProbabilitySwitch">
              <entries probability="0.5">
204             <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
206             </items>
              </entries>
208           <entries probability="0.5">
                <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
210               <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
                </items>
212           </entries>
            </items>
214         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                  =ModeLiteral" />
216         </condition>
          </entries>
218       <entries>
            <items xsi:type="am:ProbabilitySwitch">
```

```
            <entries probability="0.7">
              <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.3">
              <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
        <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ProbabilitySwitch">
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_0?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
          </items>
        </entries>
        <entries probability="30.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_4?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
```

```
276         <graphEntries xsi:type="am:CallSequence" name="CallSequence_5">
              <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_6?type=InterProcessStimulus"
                   />
278           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
            </graphEntries>
280       </callGraph>
          <customProperties key="priority">
282         <value xsi:type="am:StringObject" value="5" />
          </customProperties>
284       <customProperties key="osekTaskGroup">
            <value xsi:type="am:StringObject" value="5" />
286       </customProperties>
        </tasks>
288     <tasks name="Task_6" stimuli="IPA_Task_6?type=InterProcessStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
          <callGraph>
290         <graphEntries xsi:type="am:ModeSwitch">
              <entries>
292             <items xsi:type="am:CallSequence" name="CallSequence_6_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_1?type=Runnable" />
294             </items>
                <condition>
296               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_1?type=ModeLiteral" />
                </condition>
298           </entries>
              <entries>
300             <items xsi:type="am:CallSequence" name="CallSequence_6_2">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_2?type=Runnable" />
302             </items>
                <condition>
304               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_2?type=ModeLiteral" />
                </condition>
306           </entries>
              <entries>
308             <items xsi:type="am:CallSequence" name="CallSequence_6_3">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_3?type=Runnable" />
310             </items>
                <condition>
312               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_3?type=ModeLiteral" />
                </condition>
314           </entries>
              <entries>
316             <items xsi:type="am:CallSequence" name="CallSequence_6_4">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_4?type=Runnable" />
318             </items>
                <condition>
320               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_4?type=ModeLiteral" />
                </condition>
322           </entries>
              <defaultEntry>
324             <items xsi:type="am:CallSequence" name="CallSequence_6_x">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_x?type=Runnable" />
326             </items>
              </defaultEntry>
```

```
328         </graphEntries>
        </callGraph>
330     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="4" />
332     </customProperties>
        <customProperties key="osekTaskGroup">
334       <value xsi:type="am:StringObject" value="4" />
        </customProperties>
336   </tasks>
      <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
338     <callGraph>
          <graphEntries xsi:type="am:CallSequence" name="CS_Task_7">
340         <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
342       </graphEntries>
        </callGraph>
344   </tasks>
      <runnables name="Set_e_0" callback="false" service="false">
346     <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
            modeValue="E/E_0?type=ModeLiteral" />
      </runnables>
348   <runnables name="Set_e_1" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
            modeValue="E/E_1?type=ModeLiteral" />
350   </runnables>
      <runnables name="Set_state_0" callback="false" service="false">
352     <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
            modeValue="State/State_0?type=ModeLiteral" />
      </runnables>
354   <runnables name="Set_state_1" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
            modeValue="State/State_1?type=ModeLiteral" />
356   </runnables>
      <runnables name="Set_state_2" callback="false" service="false">
358     <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
            modeValue="State/State_2?type=ModeLiteral" />
      </runnables>
360   <runnables name="Set_y_0" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
            modeValue="Y/Y_0?type=ModeLiteral" />
362   </runnables>
      <runnables name="Set_y_50" callback="false" service="false">
364     <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
            modeValue="Y/Y_50?type=ModeLiteral" />
      </runnables>
366   <runnables name="Set_y_100" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
            modeValue="Y/Y_100?type=ModeLiteral" />
368   </runnables>
      <runnables name="R_3_0" callback="false" service="false">
370     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
372         <value xsi:type="am:NeedDeviation">
              <deviation>
374             <lowerBound xsi:type="am:LongObject" value="594000" />
                <upperBound xsi:type="am:LongObject" value="600000" />
376             <distribution xsi:type="am:UniformDistribution" />
```

```
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="R_3_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400000" />
            <upperBound xsi:type="am:LongObject" value="60000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="R_3_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Set_w_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
        modeValue="W/W_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_w_50" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
        modeValue="W/W_50?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_w_100" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
        modeValue="W/W_100?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_u_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
        modeValue="U/U_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Set_u_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
        modeValue="U/U_1?type=ModeLiteral" />
  </runnables>
  <runnables name="R1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
```

```
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="R2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594000" />
            <upperBound xsi:type="am:LongObject" value="600000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="R_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_3" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_3?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_5_4" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_4?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_6_x" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
              <upperBound xsi:type="am:LongObject" value="30000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="600" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
              <upperBound xsi:type="am:LongObject" value="30000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
```

```
    </runnables>
    <runnables name="Runnable_6_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="35640000" />
              <upperBound xsi:type="am:LongObject" value="36000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11880000" />
              <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <modes name="E">
      <literals name="E_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="E_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="U">
      <literals name="U_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
```

```
      <literals name="U_1">
600     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
602     </customProperties>
      </literals>
604   </modes>
      <modes name="Y">
606     <literals name="Y_0">
          <customProperties key="enumValue">
608         <value xsi:type="am:LongObject" value="0" />
          </customProperties>
610     </literals>
        <literals name="Y_50">
612       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="50" />
614       </customProperties>
        </literals>
616     <literals name="Y_100">
          <customProperties key="enumValue">
618         <value xsi:type="am:LongObject" value="100" />
          </customProperties>
620     </literals>
      </modes>
622   <modes name="W">
        <literals name="W_0">
624       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
626       </customProperties>
        </literals>
628     <literals name="W_50">
          <customProperties key="enumValue">
630         <value xsi:type="am:LongObject" value="50" />
          </customProperties>
632     </literals>
        <literals name="W_100">
634       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="100" />
636       </customProperties>
        </literals>
638   </modes>
      <modes name="State">
640     <literals name="State_0">
          <customProperties key="enumValue">
642         <value xsi:type="am:LongObject" value="0" />
          </customProperties>
644     </literals>
        <literals name="State_1">
646       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="1" />
648       </customProperties>
        </literals>
650     <literals name="State_2">
          <customProperties key="enumValue">
652         <value xsi:type="am:LongObject" value="2" />
          </customProperties>
654     </literals>
      </modes>
656   <modes name="Message">
```

```
      <literals name="Message_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="Message_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="Message_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
      <literals name="Message_3">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="3" />
        </customProperties>
      </literals>
      <literals name="Message_4">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="4" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
```

```
        </modules>
        <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
          <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
        </modules>
      </structures>
    </structures>
  </structures>
  <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
    <defaultValue value="600.0" unit="MHz"/>
  </domains>
</hwModel>
<osModel>
  <operatingSystems name="Generic_OS">
    <taskSchedulers name="Scheduler_1">
      <schedulingAlgorithm xsi:type="am:OSEK" />
    </taskSchedulers>
    <osDataConsistency mode="noProtection" />
  </operatingSystems>
</osModel>
<stimuliModel>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
    <offset value="0" unit="ms" />
    <recurrence value="600" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_3" />
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
    <offset value="0" unit="ms" />
    <recurrence value="100" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_6" />
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
    <offset value="0" unit="ms" />
    <recurrence value="1000" unit="ms" />
  </stimuli>
</stimuliModel>
<constraintsModel />
<eventModel>
  <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
  <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
      " />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" description="" entity="IPA_Task_2?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_3" entity="IPA_Task_3?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_6" entity="IPA_Task_6?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=PeriodicStimulus" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_0" entity="Runnable_5_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_3" entity="Runnable_5_3?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_4" entity="Runnable_5_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_1" entity="Runnable_6_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_2" entity="Runnable_6_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_3" entity="Runnable_6_3?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_4" entity="Runnable_6_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_x" entity="Runnable_6_x?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=Runnable" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
```

```
804     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
806     <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
808     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
            =ModeLabel" />
810     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
            type=ModeLabel" />
812     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
            ?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="26" abstractElement="
            message?type=ModeLabel" />
814   </mappingModel>
      <componentsModel />
816 </am:Amalthea>
```

Listing A.26: Variation 4 of Feedback Loop.

A.1.4.5. Variation 5

```
    <?xml version="1.0" encoding="UTF-8"?>
2   <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
        " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
4       <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
          <callGraph>
6           <graphEntries xsi:type="am:ModeSwitch">
              <entries>
8               <items xsi:type="am:CallSequence" name="CS_e_0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
10              </items>
                <condition>
12                <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                      ModeLiteral" />
                </condition>
14            </entries>
              <entries>
16              <items xsi:type="am:CallSequence" name="CS_e_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
18              </items>
                <condition>
20                <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                      ModeLiteral" />
                </condition>
22            </entries>
            </graphEntries>
24          <graphEntries xsi:type="am:CallSequence" name="CS_R1">
              <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
                   />
26            <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
```

```
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
```

```
80              <entries>
                  <items xsi:type="am:CallSequence" name="CS_1_to_2">
82                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                  </items>
84                <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
86                </condition>
                </entries>
88            </items>
              <condition>
90              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_1?type=ModeLiteral" />
              </condition>
92          </entries>
            <entries>
94            <items xsi:type="am:CallSequence" name="CS_state_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
96              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
              </items>
98            <items xsi:type="am:ModeSwitch">
                <entries>
100               <items xsi:type="am:CallSequence" name="CS_2_to_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
102               </items>
                  <condition>
104                 <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                        type=ModeLiteral" />
                  </condition>
106             </entries>
                <entries>
108               <items xsi:type="am:CallSequence" name="CS_2_to_2">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
110               </items>
                  <condition>
112                 <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                        type=ModeLiteral" />
                  </condition>
114             </entries>
              </items>
116           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                    State_2?type=ModeLiteral" />
118           </condition>
            </entries>
120       </graphEntries>
          <graphEntries xsi:type="am:CallSequence" name="CS_IPA_T3">
122         <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_3?type=InterProcessStimulus"
                />
          </graphEntries>
124       <graphEntries xsi:type="am:ProbabilitySwitch">
            <entries probability="0.3">
126           <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
                <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                    InterProcessStimulus" />
128           </items>
            </entries>
130         <entries probability="0.7">
```

```
          <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_R2">
        <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_3" stimuli="IPA_Task_3?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                =ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
```

```
184           <entries>
              <items xsi:type="am:ProbabilitySwitch">
186             <entries probability="0.3">
                  <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
188                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                  </items>
190             </entries>
                <entries probability="0.7">
192               <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
194               </items>
                </entries>
196           </items>
              <condition>
198             <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                    ModeLiteral" />
              </condition>
200         </entries>
            <entries>
202           <items xsi:type="am:ProbabilitySwitch">
                <entries probability="0.5">
204               <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
206               </items>
                </entries>
208             <entries probability="0.5">
                  <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
210                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
                  </items>
212             </entries>
              </items>
214           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                    =ModeLiteral" />
216           </condition>
            </entries>
218         <entries>
              <items xsi:type="am:ProbabilitySwitch">
220             <entries probability="0.7">
                  <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
222                 <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                  </items>
224             </entries>
                <entries probability="0.3">
226               <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
228               </items>
                </entries>
230           </items>
              <condition>
232             <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                    type=ModeLiteral" />
              </condition>
234         </entries>
          </graphEntries>
236       <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
238       </graphEntries>
```

```
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
      </customProperties>
    </tasks>
    <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_5_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_0?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_5_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_5_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_5_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_5_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_4?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_5">
          <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_6?type=InterProcessStimulus"
              />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="5" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="5" />
      </customProperties>
    </tasks>
    <tasks name="Task_6" stimuli="IPA_Task_6?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_6_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_1?type=Runnable" />
```

```
294             </items>
                <condition>
296               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_1?type=ModeLiteral" />
                </condition>
298           </entries>
              <entries>
300             <items xsi:type="am:CallSequence" name="CallSequence_6_2">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_2?type=Runnable" />
302             </items>
                <condition>
304               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_2?type=ModeLiteral" />
                </condition>
306           </entries>
              <entries>
308             <items xsi:type="am:CallSequence" name="CallSequence_6_3">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_3?type=Runnable" />
310             </items>
                <condition>
312               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_3?type=ModeLiteral" />
                </condition>
314           </entries>
              <entries>
316             <items xsi:type="am:CallSequence" name="CallSequence_6_4">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_4?type=Runnable" />
318             </items>
                <condition>
320               <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_4?type=ModeLiteral" />
                </condition>
322           </entries>
              <defaultEntry>
324             <items xsi:type="am:CallSequence" name="CallSequence_6_x">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_x?type=Runnable" />
326             </items>
              </defaultEntry>
328         </graphEntries>
          </callGraph>
330       <customProperties key="priority">
            <value xsi:type="am:StringObject" value="4" />
332       </customProperties>
          <customProperties key="osekTaskGroup">
334         <value xsi:type="am:StringObject" value="4" />
          </customProperties>
336     </tasks>
        <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
338       <callGraph>
            <graphEntries xsi:type="am:CallSequence" name="CS_Task_7">
340           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
342         </graphEntries>
          </callGraph>
344     </tasks>
        <runnables name="Set_e_0" callback="false" service="false">
```

```
346       <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
              modeValue="E/E_0?type=ModeLiteral" />
          </runnables>
348       <runnables name="Set_e_1" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
              modeValue="E/E_1?type=ModeLiteral" />
350       </runnables>
          <runnables name="Set_state_0" callback="false" service="false">
352         <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_0?type=ModeLiteral" />
          </runnables>
354       <runnables name="Set_state_1" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_1?type=ModeLiteral" />
356       </runnables>
          <runnables name="Set_state_2" callback="false" service="false">
358         <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
              modeValue="State/State_2?type=ModeLiteral" />
          </runnables>
360       <runnables name="Set_y_0" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
              modeValue="Y/Y_0?type=ModeLiteral" />
362       </runnables>
          <runnables name="Set_y_50" callback="false" service="false">
364         <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
              modeValue="Y/Y_50?type=ModeLiteral" />
          </runnables>
366       <runnables name="Set_y_100" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
              modeValue="Y/Y_100?type=ModeLiteral" />
368       </runnables>
          <runnables name="R_3_0" callback="false" service="false">
370         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
372             <value xsi:type="am:NeedDeviation">
                  <deviation>
374                 <lowerBound xsi:type="am:LongObject" value="594000" />
                    <upperBound xsi:type="am:LongObject" value="600000" />
376                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
378             </value>
              </default>
380         </runnableItems>
          </runnables>
382       <runnables name="R_3_2" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
384           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
386               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="59400000" />
388                 <upperBound xsi:type="am:LongObject" value="60000000" />
                    <distribution xsi:type="am:UniformDistribution" />
390               </deviation>
                </value>
392           </default>
            </runnableItems>
394       </runnables>
          <runnables name="R_3_1" callback="false" service="false">
```

```
396       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
398           <value xsi:type="am:NeedDeviation">
                <deviation>
400               <lowerBound xsi:type="am:LongObject" value="5940000" />
                  <upperBound xsi:type="am:LongObject" value="6000000" />
402               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
404           </value>
            </default>
406       </runnableItems>
        </runnables>
408     <runnables name="Set_w_0" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_0?type=ModeLiteral" />
410     </runnables>
        <runnables name="Set_w_50" callback="false" service="false">
412       <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_50?type=ModeLiteral" />
        </runnables>
414     <runnables name="Set_w_100" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_100?type=ModeLiteral" />
416     </runnables>
        <runnables name="Set_u_0" callback="false" service="false">
418       <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
              modeValue="U/U_0?type=ModeLiteral" />
        </runnables>
420     <runnables name="Set_u_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
              modeValue="U/U_1?type=ModeLiteral" />
422     </runnables>
        <runnables name="R1" callback="false" service="false">
424       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
426           <value xsi:type="am:NeedDeviation">
                <deviation>
428               <lowerBound xsi:type="am:LongObject" value="5940000" />
                  <upperBound xsi:type="am:LongObject" value="6000000" />
430               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
432           </value>
            </default>
434       </runnableItems>
        </runnables>
436     <runnables name="R2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
438         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
440             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="594000" />
442               <upperBound xsi:type="am:LongObject" value="600000" />
                  <distribution xsi:type="am:UniformDistribution" />
444             </deviation>
              </value>
446         </default>
          </runnableItems>
448     </runnables>
```

```
    <runnables name="R_4" callback="false" service="false">
450   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
452       <value xsi:type="am:NeedDeviation">
            <deviation>
454           <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
456           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
458       </value>
        </default>
460   </runnableItems>
    </runnables>
462 <runnables name="Runnable_5" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
464     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
466         <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
468           <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
470         </deviation>
          </value>
472     </default>
      </runnableItems>
474 </runnables>
    <runnables name="Runnable_5_0" callback="false" service="false">
476   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_0?type=ModeLiteral" />
    </runnables>
478 <runnables name="Runnable_5_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_1?type=ModeLiteral" />
480 </runnables>
    <runnables name="Runnable_5_2" callback="false" service="false">
482   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_2?type=ModeLiteral" />
    </runnables>
484 <runnables name="Runnable_5_3" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_3?type=ModeLiteral" />
486 </runnables>
    <runnables name="Runnable_5_4" callback="false" service="false">
488   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_4?type=ModeLiteral" />
    </runnables>
490 <runnables name="Runnable_6_x" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
492     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
494         <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
496           <upperBound xsi:type="am:LongObject" value="30000000" />
              <distribution xsi:type="am:UniformDistribution" />
498         </deviation>
          </value>
500     </default>
      </runnableItems>
```

```
502     </runnables>
        <runnables name="Runnable_6_1" callback="false" service="false">
504       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
506           <value xsi:type="am:NeedDeviation">
                <deviation>
508               <lowerBound xsi:type="am:LongObject" value="5940000" />
                  <upperBound xsi:type="am:LongObject" value="6000000" />
510               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
512           </value>
            </default>
514       </runnableItems>
        </runnables>
516     <runnables name="Runnable_6_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
518         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
520             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="594" />
522               <upperBound xsi:type="am:LongObject" value="600" />
                  <distribution xsi:type="am:UniformDistribution" />
524             </deviation>
              </value>
526         </default>
          </runnableItems>
528     </runnables>
        <runnables name="Runnable_6_3" callback="false" service="false">
530       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
532           <value xsi:type="am:NeedDeviation">
                <deviation>
534               <lowerBound xsi:type="am:LongObject" value="29700" />
                  <upperBound xsi:type="am:LongObject" value="30000" />
536               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
538           </value>
            </default>
540       </runnableItems>
        </runnables>
542     <runnables name="Runnable_6_4" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
544         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
546             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="594000" />
548               <upperBound xsi:type="am:LongObject" value="600000" />
                  <distribution xsi:type="am:UniformDistribution" />
550             </deviation>
              </value>
552         </default>
          </runnableItems>
554     </runnables>
        <runnables name="Runnable_7_1" callback="false" service="false">
556       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
558           <value xsi:type="am:NeedDeviation">
                <deviation>
```

```
560             <lowerBound xsi:type="am:LongObject" value="35640000" />
                <upperBound xsi:type="am:LongObject" value="36000000" />
562             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
564         </value>
          </default>
566     </runnableItems>
    </runnables>
568 <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
570       <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
572         <deviation>
              <lowerBound xsi:type="am:LongObject" value="11880000" />
574           <upperBound xsi:type="am:LongObject" value="12000000" />
              <distribution xsi:type="am:UniformDistribution" />
576         </deviation>
          </value>
578       </default>
      </runnableItems>
580 </runnables>
    <modes name="E">
582   <literals name="E_0">
        <customProperties key="enumValue">
584       <value xsi:type="am:LongObject" value="0" />
        </customProperties>
586   </literals>
      <literals name="E_1">
588     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
590     </customProperties>
      </literals>
592 </modes>
    <modes name="U">
594   <literals name="U_0">
        <customProperties key="enumValue">
596       <value xsi:type="am:LongObject" value="0" />
        </customProperties>
598   </literals>
      <literals name="U_1">
600     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
602     </customProperties>
      </literals>
604 </modes>
    <modes name="Y">
606   <literals name="Y_0">
        <customProperties key="enumValue">
608       <value xsi:type="am:LongObject" value="0" />
        </customProperties>
610   </literals>
      <literals name="Y_50">
612     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="50" />
614     </customProperties>
      </literals>
616   <literals name="Y_100">
        <customProperties key="enumValue">
```

```
618           <value xsi:type="am:LongObject" value="100" />
          </customProperties>
620       </literals>
      </modes>
622     <modes name="W">
        <literals name="W_0">
624       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
626       </customProperties>
        </literals>
628     <literals name="W_50">
          <customProperties key="enumValue">
630         <value xsi:type="am:LongObject" value="50" />
          </customProperties>
632     </literals>
        <literals name="W_100">
634       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="100" />
636       </customProperties>
        </literals>
638   </modes>
      <modes name="State">
640     <literals name="State_0">
          <customProperties key="enumValue">
642         <value xsi:type="am:LongObject" value="0" />
          </customProperties>
644     </literals>
        <literals name="State_1">
646       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="1" />
648       </customProperties>
        </literals>
650     <literals name="State_2">
          <customProperties key="enumValue">
652         <value xsi:type="am:LongObject" value="2" />
          </customProperties>
654     </literals>
      </modes>
656   <modes name="Message">
        <literals name="Message_0">
658       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="0" />
660       </customProperties>
        </literals>
662     <literals name="Message_1">
          <customProperties key="enumValue">
664         <value xsi:type="am:LongObject" value="1" />
          </customProperties>
666     </literals>
        <literals name="Message_2">
668       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="2" />
670       </customProperties>
        </literals>
672     <literals name="Message_3">
          <customProperties key="enumValue">
674         <value xsi:type="am:LongObject" value="3" />
          </customProperties>
```

```
676       </literals>
          <literals name="Message_4">
678         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="4" />
680         </customProperties>
          </literals>
682     </modes>
        <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
684       <size value="1" unit="bit" />
        </modeLabels>
686     <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
          <size value="8" unit="bit" />
688     </modeLabels>
        <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
690       <size value="8" unit="bit" />
        </modeLabels>
692     <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
          <size value="8" unit="bit" />
694     </modeLabels>
        <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
696       <size value="1" unit="bit" />
        </modeLabels>
698     <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
          <size value="8" unit="bit" />
700     </modeLabels>
      </swModel>
702   <hwModel>
        <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
704     <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
        </definitions>
706     <featureCategories name="Instructions" featureType="performance">
          <features name="IPC_1.0" value="1.0" />
708     </featureCategories>
        <structures name="System" structureType="System">
710       <structures name="Ecu_1" structureType="ECU">
            <structures name="Processor_1" structureType="Microcontroller">
712           <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
              </modules>
714           <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
                <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
716           </modules>
            </structures>
718       </structures>
        </structures>
720     <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
          <defaultValue value="600.0" unit="MHz"/>
722     </domains>
      </hwModel>
724   <osModel>
        <operatingSystems name="Generic_OS">
726       <taskSchedulers name="Scheduler_1">
            <schedulingAlgorithm xsi:type="am:OSEK" />
728       </taskSchedulers>
          <osDataConsistency mode="noProtection" />
730     </operatingSystems>
```

```
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="450" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_3" />
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="60" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_6" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
      <recurrence value="575" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" description="" entity="IPA_Task_2?
        type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_3" entity="IPA_Task_3?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
```

```
782     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_6" entity="IPA_Task_6?type=
            InterProcessStimulus" />
784     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
            PeriodicStimulus" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
            />
786     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_0" entity="Runnable_5_0?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
            Runnable" />
788     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_3" entity="Runnable_5_3?type=
            Runnable" />
790     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_4" entity="Runnable_5_4?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_1" entity="Runnable_6_1?type=
            Runnable" />
792     <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_2" entity="Runnable_6_2?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_3" entity="Runnable_6_3?type=
            Runnable" />
794     <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_4" entity="Runnable_6_4?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_x" entity="Runnable_6_x?type=
            Runnable" />
796     <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
            Runnable" />
798   </eventModel>
      <mappingModel addressMappingType="offset">
800     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
802     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
804     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
806     <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
808     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
            =ModeLabel" />
810     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
            type=ModeLabel" />
812     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
            ?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="26" abstractElement="
            message?type=ModeLabel" />
814   </mappingModel>
      <componentsModel />
816 </am:Amalthea>
```

Listing A.27: Variation 5 of Feedback Loop.

A.1.4.6. Variation 6

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_e_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                  ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_e_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                  ModeLiteral" />
            </condition>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_R1">
          <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
              />
          <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
    </tasks>
    <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_state_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
              <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
            </items>
            <items xsi:type="am:ModeSwitch">
```

```
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
```

```
            </items>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CS_2_to_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                      type=ModeLiteral" />
                </condition>
              </entries>
              <entries>
                <items xsi:type="am:CallSequence" name="CS_2_to_2">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                      type=ModeLiteral" />
                </condition>
              </entries>
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_2?type=ModeLiteral" />
            </condition>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_IPA_T3">
          <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_3?type=InterProcessStimulus"
               />
        </graphEntries>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="0.3">
            <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
              <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                  InterProcessStimulus" />
            </items>
          </entries>
          <entries probability="0.7">
            <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_R2">
          <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <tasks name="Task_3" stimuli="IPA_Task_3?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
```

```
          <items xsi:type="am:CallSequence" name="CS_y_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                =ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_y_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.3">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.7">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
```

```
202             <items xsi:type="am:ProbabilitySwitch">
                  <entries probability="0.5">
204                 <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
206                 </items>
                  </entries>
208               <entries probability="0.5">
                    <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
210                   <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
                    </items>
212               </entries>
                </items>
214             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                      =ModeLiteral" />
216             </condition>
              </entries>
218           <entries>
                <items xsi:type="am:ProbabilitySwitch">
220               <entries probability="0.7">
                    <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
222                   <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
                    </items>
224               </entries>
                  <entries probability="0.3">
226                 <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
228                 </items>
                  </entries>
230             </items>
                <condition>
232               <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                      type=ModeLiteral" />
                </condition>
234           </entries>
          </graphEntries>
236       <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
            <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
238       </graphEntries>
        </callGraph>
240     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
242     </customProperties>
        <customProperties key="osekTaskGroup">
244       <value xsi:type="am:StringObject" value="1" />
        </customProperties>
246   </tasks>
      <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
248     <callGraph>
          <graphEntries xsi:type="am:ProbabilitySwitch">
250         <entries probability="15.0">
              <items xsi:type="am:CallSequence" name="CallSequence_5_0">
252             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_0?type=Runnable" />
              </items>
254         </entries>
            <entries probability="20.0">
256           <items xsi:type="am:CallSequence" name="CallSequence_5_1">
```

```
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
          </items>
        </entries>
        <entries probability="30.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_4?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_6?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
    </customProperties>
  </tasks>
  <tasks name="Task_6" stimuli="IPA_Task_6?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_6_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_3?type=Runnable" />
          </items>
```

```
            <condition>
312           <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_3?type=ModeLiteral" />
            </condition>
314       </entries>
          <entries>
316         <items xsi:type="am:CallSequence" name="CallSequence_6_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_4?type=Runnable" />
318         </items>
            <condition>
320           <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_4?type=ModeLiteral" />
            </condition>
322       </entries>
          <defaultEntry>
324         <items xsi:type="am:CallSequence" name="CallSequence_6_x">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_x?type=Runnable" />
326         </items>
          </defaultEntry>
328     </graphEntries>
      </callGraph>
330   <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
332   </customProperties>
      <customProperties key="osekTaskGroup">
334     <value xsi:type="am:StringObject" value="4" />
      </customProperties>
336 </tasks>
    <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
338   <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CS_Task_7">
340       <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
342     </graphEntries>
      </callGraph>
344 </tasks>
    <runnables name="Set_e_0" callback="false" service="false">
346   <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_0?type=ModeLiteral" />
    </runnables>
348 <runnables name="Set_e_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_1?type=ModeLiteral" />
350 </runnables>
    <runnables name="Set_state_0" callback="false" service="false">
352   <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
    </runnables>
354 <runnables name="Set_state_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
356 </runnables>
    <runnables name="Set_state_2" callback="false" service="false">
358   <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
360 <runnables name="Set_y_0" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_50?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_100" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_100?type=ModeLiteral" />
    </runnables>
    <runnables name="R_3_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="606000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400000" />
              <upperBound xsi:type="am:LongObject" value="60600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Set_w_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_50?type=ModeLiteral" />
    </runnables>
```

```
414     <runnables name="Set_w_100" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
              modeValue="W/W_100?type=ModeLiteral" />
416     </runnables>
        <runnables name="Set_u_0" callback="false" service="false">
418       <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
              modeValue="U/U_0?type=ModeLiteral" />
        </runnables>
420     <runnables name="Set_u_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
              modeValue="U/U_1?type=ModeLiteral" />
422     </runnables>
        <runnables name="R1" callback="false" service="false">
424       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
426           <value xsi:type="am:NeedDeviation">
                <deviation>
428               <lowerBound xsi:type="am:LongObject" value="5940000" />
                  <upperBound xsi:type="am:LongObject" value="6060000" />
430               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
432           </value>
            </default>
434       </runnableItems>
        </runnables>
436     <runnables name="R2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
438         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
440             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="594000" />
442               <upperBound xsi:type="am:LongObject" value="606000" />
                  <distribution xsi:type="am:UniformDistribution" />
444             </deviation>
              </value>
446         </default>
          </runnableItems>
448     </runnables>
        <runnables name="R_4" callback="false" service="false">
450       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
452           <value xsi:type="am:NeedDeviation">
                <deviation>
454               <lowerBound xsi:type="am:LongObject" value="5940000" />
                  <upperBound xsi:type="am:LongObject" value="6060000" />
456               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
458           </value>
            </default>
460       </runnableItems>
        </runnables>
462     <runnables name="Runnable_5" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
464         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
466             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="5940000" />
468               <upperBound xsi:type="am:LongObject" value="6060000" />
```

```
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_3" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_3?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_4" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_4?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_6_x" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700000" />
            <upperBound xsi:type="am:LongObject" value="30300000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_6_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_6_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594" />
```

```
            <upperBound xsi:type="am:LongObject" value="606" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_6_3" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700" />
            <upperBound xsi:type="am:LongObject" value="30300" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_6_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594000" />
            <upperBound xsi:type="am:LongObject" value="606000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_7_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="35640000" />
            <upperBound xsi:type="am:LongObject" value="36360000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_7_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="11880000" />
            <upperBound xsi:type="am:LongObject" value="12120000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
```

```
580     </runnables>
        <modes name="E">
582       <literals name="E_0">
            <customProperties key="enumValue">
584           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
586       </literals>
          <literals name="E_1">
588         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
590         </customProperties>
          </literals>
592     </modes>
        <modes name="U">
594       <literals name="U_0">
            <customProperties key="enumValue">
596           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
598       </literals>
          <literals name="U_1">
600         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
602         </customProperties>
          </literals>
604     </modes>
        <modes name="Y">
606       <literals name="Y_0">
            <customProperties key="enumValue">
608           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
610       </literals>
          <literals name="Y_50">
612         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="50" />
614         </customProperties>
          </literals>
616       <literals name="Y_100">
            <customProperties key="enumValue">
618           <value xsi:type="am:LongObject" value="100" />
            </customProperties>
620       </literals>
        </modes>
622     <modes name="W">
          <literals name="W_0">
624         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
626         </customProperties>
          </literals>
628       <literals name="W_50">
            <customProperties key="enumValue">
630           <value xsi:type="am:LongObject" value="50" />
            </customProperties>
632       </literals>
          <literals name="W_100">
634         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="100" />
636         </customProperties>
          </literals>
```

```
638     </modes>
        <modes name="State">
640       <literals name="State_0">
            <customProperties key="enumValue">
642           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
644       </literals>
          <literals name="State_1">
646         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
648         </customProperties>
          </literals>
650       <literals name="State_2">
            <customProperties key="enumValue">
652           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
654       </literals>
        </modes>
656     <modes name="Message">
          <literals name="Message_0">
658         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
660         </customProperties>
          </literals>
662       <literals name="Message_1">
            <customProperties key="enumValue">
664           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
666       </literals>
          <literals name="Message_2">
668         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
670         </customProperties>
          </literals>
672       <literals name="Message_3">
            <customProperties key="enumValue">
674           <value xsi:type="am:LongObject" value="3" />
            </customProperties>
676       </literals>
          <literals name="Message_4">
678         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="4" />
680         </customProperties>
          </literals>
682     </modes>
        <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
684       <size value="1" unit="bit" />
        </modeLabels>
686     <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
          <size value="8" unit="bit" />
688     </modeLabels>
        <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
690       <size value="8" unit="bit" />
        </modeLabels>
692     <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
          <size value="8" unit="bit" />
694     </modeLabels>
        <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
```

```
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="450" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_3" />
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="60" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_6" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
      <offset value="0" unit="ms" />
      <recurrence value="575" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
```

```
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable
        " />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" description="" entity="IPA_Task_2?
        type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_3" entity="IPA_Task_3?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_6" entity="IPA_Task_6?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=
        PeriodicStimulus" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_0" entity="Runnable_5_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_3" entity="Runnable_5_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_4" entity="Runnable_5_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_1" entity="Runnable_6_1?type=
        Runnable" />
```

```
792     <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_2" entity="Runnable_6_2?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_3" entity="Runnable_6_3?type=
            Runnable" />
794     <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_4" entity="Runnable_6_4?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_x" entity="Runnable_6_x?type=
            Runnable" />
796     <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=
            Runnable" />
798   </eventModel>
    <mappingModel addressMappingType="offset">
800     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
802     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
804     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
806     <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
808     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
            =ModeLabel" />
810     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
            type=ModeLabel" />
812     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
            ?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="26" abstractElement="
            message?type=ModeLabel" />
814   </mappingModel>
    <componentsModel />
816 </am:Amalthea>
```

Listing A.28: Variation 6 of Feedback Loop.

A.1.4.7. Variation 7

```
    <?xml version="1.0" encoding="UTF-8"?>
2   <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
        " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
4       <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="2">
          <callGraph>
6           <graphEntries xsi:type="am:ModeSwitch">
              <entries>
8               <items xsi:type="am:CallSequence" name="CS_e_0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_0?type=Runnable" />
10              </items>
                <condition>
```

```
            <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_e_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_u_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="e?type=ModeLabel" value="E/E_1?type=
                ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_R1">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="R1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_0?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_0_to_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
```

```
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_50?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_50?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_0?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_1_to_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                State_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CS_state_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_y_100?type=Runnable" />
            <calls xsi:type="am:TaskRunnableCall" runnable="Set_w_100?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CS_2_to_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_1?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_0?
                    type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CS_2_to_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_state_2?type=Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="u?type=ModeLabel" value="U/U_1?
                    type=ModeLiteral" />
              </condition>
            </entries>
```

```
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="state?type=ModeLabel" value="State/
                  State_2?type=ModeLiteral" />
            </condition>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_IPA_T3">
          <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_3?type=InterProcessStimulus"
               />
        </graphEntries>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="0.3">
            <items xsi:type="am:CallSequence" name="CS_Trigger_Task_4">
              <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=
                  InterProcessStimulus" />
            </items>
          </entries>
          <entries probability="0.7">
            <items xsi:type="am:CallSequence" name="CS_w_notrigger" />
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CS_R2">
          <calls xsi:type="am:TaskRunnableCall" runnable="R2?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="2" />
      </customProperties>
    </tasks>
    <tasks name="Task_3" stimuli="IPA_Task_3?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CS_y_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="R_3_0?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_0?type=
                  ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_y_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="R_3_1?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_50?type
                  =ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CS_y_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="R_3_2?type=Runnable" />
```

```
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="y?type=ModeLabel" value="Y/Y_100?
                type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.3">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.7">
              <items xsi:type="am:CallSequence" name="CS_w_0_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_0?type=
                ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.5">
              <items xsi:type="am:CallSequence" name="CS_w_50_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.5">
              <items xsi:type="am:CallSequence" name="CS_w_50_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_50?type
                =ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:ProbabilitySwitch">
            <entries probability="0.7">
```

```
              <items xsi:type="am:CallSequence" name="CS_w_100_e_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_0?type=Runnable" />
              </items>
            </entries>
            <entries probability="0.3">
              <items xsi:type="am:CallSequence" name="CS_w_100_e_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Set_e_1?type=Runnable" />
              </items>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="w?type=ModeLabel" value="W/W_100?
                type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CS_Task_4_Post">
        <calls xsi:type="am:TaskRunnableCall" runnable="R_4?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ProbabilitySwitch">
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_0?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
          </items>
        </entries>
        <entries probability="30.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_5_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_4?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_5">
```

```
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_6?type=InterProcessStimulus"
            />
278     <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5?type=Runnable" />
      </graphEntries>
280   </callGraph>
    <customProperties key="priority">
282   <value xsi:type="am:StringObject" value="5" />
    </customProperties>
284 <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="5" />
286 </customProperties>
  </tasks>
288 <tasks name="Task_6" stimuli="IPA_Task_6?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
290   <graphEntries xsi:type="am:ModeSwitch">
        <entries>
292       <items xsi:type="am:CallSequence" name="CallSequence_6_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_1?type=Runnable" />
294       </items>
          <condition>
296         <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_1?type=ModeLiteral" />
          </condition>
298     </entries>
        <entries>
300       <items xsi:type="am:CallSequence" name="CallSequence_6_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_2?type=Runnable" />
302       </items>
          <condition>
304         <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
306     </entries>
        <entries>
308       <items xsi:type="am:CallSequence" name="CallSequence_6_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_3?type=Runnable" />
310       </items>
          <condition>
312         <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_3?type=ModeLiteral" />
          </condition>
314     </entries>
        <entries>
316       <items xsi:type="am:CallSequence" name="CallSequence_6_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_4?type=Runnable" />
318       </items>
          <condition>
320         <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_4?type=ModeLiteral" />
          </condition>
322     </entries>
        <defaultEntry>
324       <items xsi:type="am:CallSequence" name="CallSequence_6_x">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_6_x?type=Runnable" />
326       </items>
        </defaultEntry>
328   </graphEntries>
```

```
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
    </tasks>
    <tasks name="Task_7" stimuli="Stimulus_Task_7?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="2">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CS_Task_7">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_1?type=Runnable" />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_7_2?type=Runnable" />
        </graphEntries>
      </callGraph>
    </tasks>
    <runnables name="Set_e_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_e_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="e?type=ModeLabel" access="write"
          modeValue="E/E_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_state_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_state_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_state_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="state?type=ModeLabel" access="write"
          modeValue="State/State_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_50?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_y_100" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="y?type=ModeLabel" access="write"
          modeValue="Y/Y_100?type=ModeLiteral" />
    </runnables>
    <runnables name="R_3_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="606000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
```

```
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400000" />
              <upperBound xsi:type="am:LongObject" value="60600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="R_3_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Set_w_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_50" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_50?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_w_100" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="w?type=ModeLabel" access="write"
          modeValue="W/W_100?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_u_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
          modeValue="U/U_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Set_u_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="u?type=ModeLabel" access="write"
          modeValue="U/U_1?type=ModeLiteral" />
    </runnables>
    <runnables name="R1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
              <distribution xsi:type="am:UniformDistribution" />
```

```
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="R2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="594000" />
            <upperBound xsi:type="am:LongObject" value="606000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="R_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_5_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_5_3" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_3?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_5_4" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_4?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_6_x" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
              <upperBound xsi:type="am:LongObject" value="30300000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6060000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="606" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_6_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
              <upperBound xsi:type="am:LongObject" value="30300" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
```

```
    </runnables>
    <runnables name="Runnable_6_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="606000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="35640000" />
              <upperBound xsi:type="am:LongObject" value="36360000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_7_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="11880000" />
              <upperBound xsi:type="am:LongObject" value="12120000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <modes name="E">
      <literals name="E_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="E_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="U">
      <literals name="U_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
```

```
      <literals name="U_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="Y">
      <literals name="Y_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="Y_50">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="50" />
        </customProperties>
      </literals>
      <literals name="Y_100">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="100" />
        </customProperties>
      </literals>
    </modes>
    <modes name="W">
      <literals name="W_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="W_50">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="50" />
        </customProperties>
      </literals>
      <literals name="W_100">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="100" />
        </customProperties>
      </literals>
    </modes>
    <modes name="State">
      <literals name="State_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="State_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="State_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
    <modes name="Message">
```

```
      <literals name="Message_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="Message_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="Message_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
      <literals name="Message_3">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="3" />
        </customProperties>
      </literals>
      <literals name="Message_4">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="4" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="e" initialValue="E/E_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="y" initialValue="Y/Y_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="w" initialValue="W/W_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="u" initialValue="U/U_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="state" initialValue="State/State_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
```

```
        </modules>
        <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
            FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
          <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
        </modules>
      </structures>
    </structures>
  </structures>
  <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
    <defaultValue value="600.0" unit="MHz"/>
  </domains>
</hwModel>
<osModel>
  <operatingSystems name="Generic_OS">
    <taskSchedulers name="Scheduler_1">
      <schedulingAlgorithm xsi:type="am:OSEK" />
    </taskSchedulers>
    <osDataConsistency mode="noProtection" />
  </operatingSystems>
</osModel>
<stimuliModel>
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
    <offset value="0" unit="ms" />
    <recurrence value="450" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_3" />
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
    <offset value="0" unit="ms" />
    <recurrence value="60" unit="ms" />
  </stimuli>
  <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_6" />
  <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_7">
    <offset value="0" unit="ms" />
    <recurrence value="575" unit="ms" />
  </stimuli>
</stimuliModel>
<constraintsModel />
<eventModel>
  <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_6" entity="Task_6?type=Task" />
  <events xsi:type="am:ProcessEvent" name="Event_Task_7" entity="Task_7?type=Task" />
  <events xsi:type="am:RunnableEvent" name="Event_R_3_0" entity="R_3_0?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R_3_1" entity="R_3_1?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R_3_2" entity="R_3_2?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R_4" entity="R_4?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R1" entity="R1?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_R2" entity="R2?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_Set_e_0" entity="Set_e_0?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_Set_e_1" entity="Set_e_1?type=Runnable" />
  <events xsi:type="am:RunnableEvent" name="Event_Set_state_0" entity="Set_state_0?type=Runnable
      " />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_1" entity="Set_state_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_state_2" entity="Set_state_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_0" entity="Set_u_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_u_1" entity="Set_u_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_0" entity="Set_w_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_50" entity="Set_w_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_w_100" entity="Set_w_100?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_0" entity="Set_y_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_50" entity="Set_y_50?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Set_y_100" entity="Set_y_100?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" description="" entity="IPA_Task_2?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_3" entity="IPA_Task_3?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_6" entity="IPA_Task_6?type=InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_7" entity="Stimulus_Task_7?type=PeriodicStimulus" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5" entity="Runnable_5?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_0" entity="Runnable_5_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_3" entity="Runnable_5_3?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_4" entity="Runnable_5_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_1" entity="Runnable_6_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_2" entity="Runnable_6_2?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_3" entity="Runnable_6_3?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_4" entity="Runnable_6_4?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_6_x" entity="Runnable_6_x?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_1" entity="Runnable_7_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_7_2" entity="Runnable_7_2?type=Runnable" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
```

```
804     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_6?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
806     <taskAllocation task="Task_7?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
808     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="u?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="1" abstractElement="e?type
            =ModeLabel" />
810     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="2" abstractElement="y?type
            =ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="w?
            type=ModeLabel" />
812     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="18" abstractElement="state
            ?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="26" abstractElement="
            message?type=ModeLabel" />
814   </mappingModel>
      <componentsModel />
816 </am:Amalthea>
```

Listing A.29: Variation 7 of Feedback Loop.

A.1.5. State Machine Feedback Loop

A.1.5.1. Variation 1

```
    <?xml version="1.0" encoding="UTF-8"?>
2   <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
        " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
4       <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
          <callGraph>
6           <graphEntries xsi:type="am:ModeSwitch">
              <entries>
8               <items xsi:type="am:ModeSwitch">
                  <entries>
10                  <items xsi:type="am:CallSequence" name="CallSequence_State0">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                           />
12                  </items>
                    <condition>
14                    <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                          StateT1/StateT1_0?type=ModeLiteral" />
                    </condition>
16                </entries>
                  <entries>
18                  <items xsi:type="am:CallSequence" name="CallSequence_State1">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                           />
20                  </items>
                    <condition>
22                    <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                          StateT1/StateT1_1?type=ModeLiteral" />
```

```
            </condition>
          </entries>
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
              MessageToT1/MessageToT1_1?type=ModeLiteral" />
        </condition>
      </entries>
      <defaultEntry>
        <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
      </defaultEntry>
    </graphEntries>
    <graphEntries xsi:type="am:ModeSwitch">
      <entries>
        <items xsi:type="am:CallSequence" name="CallSequence_1_0">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
              StateT1/StateT1_0?type=ModeLiteral" />
        </condition>
      </entries>
      <entries>
        <items xsi:type="am:CallSequence" name="CallSequence_1_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
              StateT1/StateT1_1?type=ModeLiteral" />
        </condition>
      </entries>
    </graphEntries>
    <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
      <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
    </graphEntries>
  </callGraph>
  <customProperties key="priority">
    <value xsi:type="am:StringObject" value="2" />
  </customProperties>
  <customProperties key="osekTaskGroup">
    <value xsi:type="am:StringObject" value="2" />
  </customProperties>
</tasks>
<tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
    multipleTaskActivationLimit="1">
  <callGraph>
    <graphEntries xsi:type="am:ModeSwitch">
      <entries>
        <items xsi:type="am:CallSequence" name="CallSequence_State_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
        </items>
        <items xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                  Runnable" />
            </items>
            <condition>
```

```
76                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
78          </entries>
            <entries>
80            <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                      Runnable" />
82            </items>
              <condition>
84              <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
86          </entries>
          </items>
88        <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_1?type=ModeLiteral" />
90        </condition>
        </entries>
92      <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
94          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
          </items>
96        <items xsi:type="am:ModeSwitch">
            <entries>
98            <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                      Runnable" />
100           </items>
              <condition>
102             <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
104         </entries>
            <entries>
106           <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                      Runnable" />
108           </items>
              <condition>
110             <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
112         </entries>
          </items>
114       <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_0?type=ModeLiteral" />
116       </condition>
        </entries>
118     <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_2">
120         <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
          </items>
122       <items xsi:type="am:ModeSwitch">
            <entries>
124           <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
```

```
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_2?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
  </tasks>
  <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
        " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_State_0" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59" />
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
            <upperBound xsi:type="am:LongObject" value="60000" />
            <distribution xsi:type="am:UniformDistribution" />
```

```
            </deviation>
178         </value>
          </default>
180     </runnableItems>
    </runnables>
182 <runnables name="Runnable_State_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
184     <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
186       <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700000" />
188         <upperBound xsi:type="am:LongObject" value="30000000" />
            <distribution xsi:type="am:UniformDistribution" />
190       </deviation>
        </value>
192     </default>
      </runnableItems>
194 </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
196   <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
198     <value xsi:type="am:NeedDeviation">
        <deviation>
200       <lowerBound xsi:type="am:LongObject" value="5940000" />
          <upperBound xsi:type="am:LongObject" value="6000000" />
202       <distribution xsi:type="am:UniformDistribution" />
        </deviation>
204     </value>
      </default>
206   </runnableItems>
    </runnables>
208 <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
          " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
210 </runnables>
    <runnables name="Runnable_Transition_0" callback="false" service="false">
212   <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
          modeValue="StateT2/StateT2_0?type=ModeLiteral" />
    </runnables>
214 <runnables name="Runnable_Transition_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
          modeValue="StateT2/StateT2_1?type=ModeLiteral" />
216 </runnables>
    <runnables name="Runnable_Transition_2" callback="false" service="false">
218   <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
          modeValue="StateT2/StateT2_2?type=ModeLiteral" />
    </runnables>
220 <runnables name="Runnable_1_State0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
          modeValue="StateT1/StateT1_0?type=ModeLiteral" />
222   <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
          " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
    </runnables>
224 <runnables name="Runnable_1_State1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
          modeValue="StateT1/StateT1_1?type=ModeLiteral" />
226   <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
          " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
```

```
    </runnables>
    <runnables name="Runnable_2_Overflow" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
          " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
    </runnables>
    <modes name="MessageToT1">
      <literals name="MessageToT1_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageToT1_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="MessageToT2">
      <literals name="MessageToT2_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="MessageToT2_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="StateT1">
      <literals name="StateT1_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="StateT1_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
    </modes>
    <modes name="StateT2">
      <literals name="StateT2_0">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="0" />
        </customProperties>
      </literals>
      <literals name="StateT2_1">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="StateT2_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
```

```
284     <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
          <size value="1" unit="bit" />
286     </modeLabels>
        <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
288       <size value="1" unit="bit" />
        </modeLabels>
290     <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
          <size value="1" unit="bit" />
292     </modeLabels>
        <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
294       <size value="8" unit="bit" />
        </modeLabels>
296   </swModel>
      <hwModel>
298     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
300     </definitions>
        <featureCategories name="Instructions" featureType="performance">
302       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
304     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
306         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
308           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
310             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
312         </structures>
          </structures>
314     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
316       <defaultValue value="600.0" unit="MHz"/>
        </domains>
318   </hwModel>
      <osModel>
320     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
322         <schedulingAlgorithm xsi:type="am:OSEK" />
          </taskSchedulers>
324       <osDataConsistency mode="noProtection" />
        </operatingSystems>
326   </osModel>
      <stimuliModel>
328     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
          <offset value="0" unit="ms" />
330       <recurrence value="300" unit="ms" />
        </stimuli>
332     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
          <offset value="15" unit="ms" />
334       <recurrence value="250" unit="ms" />
        </stimuli>
336   </stimuliModel>
      <constraintsModel />
338   <eventModel>
```

```
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
        Runnable_2_Overflow?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
        stateT2?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
        stateT1?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        messageToT1?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
        messageToT2?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.30: Variation 1 of State Machine Feedback Loop.

A.1.5.2. Variation 2

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
```

```
<tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
    multipleTaskActivationLimit="1">
  <callGraph>
    <graphEntries xsi:type="am:ModeSwitch">
      <entries>
        <items xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_State0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                  />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                  StateT1/StateT1_0?type=ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_State1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                  />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                  StateT1/StateT1_1?type=ModeLiteral" />
            </condition>
          </entries>
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
              MessageToT1/MessageToT1_1?type=ModeLiteral" />
        </condition>
      </entries>
      <defaultEntry>
        <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
      </defaultEntry>
    </graphEntries>
    <graphEntries xsi:type="am:ModeSwitch">
      <entries>
        <items xsi:type="am:CallSequence" name="CallSequence_1_0">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
              StateT1/StateT1_0?type=ModeLiteral" />
        </condition>
      </entries>
      <entries>
        <items xsi:type="am:CallSequence" name="CallSequence_1_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
              StateT1/StateT1_1?type=ModeLiteral" />
        </condition>
      </entries>
    </graphEntries>
    <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
      <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
```

```
54        </graphEntries>
        </callGraph>
56      <customProperties key="priority">
          <value xsi:type="am:StringObject" value="2" />
58      </customProperties>
        <customProperties key="osekTaskGroup">
60        <value xsi:type="am:StringObject" value="2" />
        </customProperties>
62    </tasks>
      <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
64      <callGraph>
          <graphEntries xsi:type="am:ModeSwitch">
66          <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State_1">
68              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
              </items>
70            <items xsi:type="am:ModeSwitch">
                <entries>
72                <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                        Runnable" />
74                </items>
                  <condition>
76                  <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
                  </condition>
78              </entries>
                <entries>
80                <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                        Runnable" />
82                </items>
                  <condition>
84                  <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
                  </condition>
86              </entries>
              </items>
88            <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_1?type=ModeLiteral" />
90            </condition>
            </entries>
92          <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State_0">
94              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
              </items>
96            <items xsi:type="am:ModeSwitch">
                <entries>
98                <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                        Runnable" />
100               </items>
                  <condition>
102                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
                  </condition>
```

```
104             </entries>
                <entries>
106               <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                        Runnable" />
108               </items>
                  <condition>
110                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
                  </condition>
112             </entries>
              </items>
114           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_0?type=ModeLiteral" />
116           </condition>
            </entries>
118         <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State_2">
120             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
              </items>
122           <items xsi:type="am:ModeSwitch">
                <entries>
124               <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                        Runnable" />
126               </items>
                  <condition>
128                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
                  </condition>
130             </entries>
                <entries>
132               <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                        Runnable" />
134               </items>
                  <condition>
136                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
                  </condition>
138             </entries>
              </items>
140           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_2?type=ModeLiteral" />
142           </condition>
            </entries>
144       </graphEntries>
      </callGraph>
146   <customProperties key="priority">
        <value xsi:type="am:StringObject" value="1" />
148   </customProperties>
      <customProperties key="osekTaskGroup">
150     <value xsi:type="am:StringObject" value="1" />
      </customProperties>
152 </tasks>
```

```
    <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_3?type=Runnable" />
            </items>
          </entries>
          <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_2?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_4?type=Runnable" />
            </items>
          </entries>
          <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_1?type=Runnable" />
            </items>
          </entries>
          <entries probability="15.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_0?type=Runnable" />
            </items>
          </entries>
        </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_3">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="4" />
      </customProperties>
    </tasks>
    <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_2?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_2?type=ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_3?type=Runnable" />
            </items>
```

```
208           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    Message/Message_3?type=ModeLiteral" />
210           </condition>
            </entries>
212         <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_4_1">
214             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_1?type=Runnable" />
              </items>
216           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    Message/Message_1?type=ModeLiteral" />
218           </condition>
            </entries>
220         <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_4_4">
222             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_4?type=Runnable" />
              </items>
224           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    Message/Message_4?type=ModeLiteral" />
226           </condition>
            </entries>
228         <defaultEntry>
              <items xsi:type="am:CallSequence" name="CallSequence_4_x">
230             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_x?type=Runnable" />
              </items>
232           </defaultEntry>
          </graphEntries>
234     </callGraph>
        <customProperties key="priority">
236       <value xsi:type="am:StringObject" value="3" />
        </customProperties>
238     <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="3" />
240     </customProperties>
      </tasks>
242   <runnables name="Runnable_1_1" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
            " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
244   </runnables>
      <runnables name="Runnable_State_0" callback="false" service="false">
246     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
248         <value xsi:type="am:NeedDeviation">
              <deviation>
250             <lowerBound xsi:type="am:LongObject" value="59" />
                <upperBound xsi:type="am:LongObject" value="60" />
252             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
254         </value>
          </default>
256     </runnableItems>
      </runnables>
258   <runnables name="Runnable_State_1" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
260       <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
```

```
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59400" />
              <upperBound xsi:type="am:LongObject" value="60000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_State_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700000" />
              <upperBound xsi:type="am:LongObject" value="30000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
              <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_1_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
          " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
          modeValue="StateT2/StateT2_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
          modeValue="StateT2/StateT2_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_Transition_2" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
          modeValue="StateT2/StateT2_2?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_State0" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
          modeValue="StateT1/StateT1_0?type=ModeLiteral" />
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
          " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_1_State1" callback="false" service="false">
```

```
314       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_1?type=ModeLiteral" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
316     </runnables>
        <runnables name="Runnable_2_Overflow" callback="false" service="false">
318       <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
        </runnables>
320     <runnables name="Runnable_3_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_1?type=ModeLiteral" />
322     </runnables>
        <runnables name="Runnable_4_1" callback="false" service="false">
324       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
326           <value xsi:type="am:NeedDeviation">
                <deviation>
328               <lowerBound xsi:type="am:LongObject" value="594" />
                  <upperBound xsi:type="am:LongObject" value="600" />
330               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
332           </value>
            </default>
334       </runnableItems>
        </runnables>
336     <runnables name="Runnable_4_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
338         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
340             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700" />
342               <upperBound xsi:type="am:LongObject" value="30000" />
                  <distribution xsi:type="am:UniformDistribution" />
344             </deviation>
              </value>
346         </default>
          </runnableItems>
348     </runnables>
        <runnables name="Runnable_4_3" callback="false" service="false">
350       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
352           <value xsi:type="am:NeedDeviation">
                <deviation>
354               <lowerBound xsi:type="am:LongObject" value="594000" />
                  <upperBound xsi:type="am:LongObject" value="600000" />
356               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
358           </value>
            </default>
360       </runnableItems>
        </runnables>
362     <runnables name="Runnable_4_4" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
364         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
366             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="23760000" />
```

```
368             <upperBound xsi:type="am:LongObject" value="24000000" />
                <distribution xsi:type="am:UniformDistribution" />
370           </deviation>
            </value>
372       </default>
        </runnableItems>
374   </runnables>
      <runnables name="Runnable_4_x" callback="false" service="false">
376     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
378         <value xsi:type="am:NeedDeviation">
              <deviation>
380             <lowerBound xsi:type="am:LongObject" value="59" />
                <upperBound xsi:type="am:LongObject" value="60" />
382             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
384         </value>
          </default>
386     </runnableItems>
      </runnables>
388   <runnables name="Runnable_3_2" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/Message_2?type=ModeLiteral" />
390   </runnables>
      <runnables name="Runnable_3_3" callback="false" service="false">
392     <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/Message_3?type=ModeLiteral" />
      </runnables>
394   <runnables name="Runnable_3_4" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/Message_4?type=ModeLiteral" />
396   </runnables>
      <runnables name="Runnable_3_0" callback="false" service="false">
398     <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/Message_0?type=ModeLiteral" />
      </runnables>
400   <runnables name="Runnable_3" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
402       <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
404           <deviation>
                <lowerBound xsi:type="am:LongObject" value="5940000" />
406             <upperBound xsi:type="am:LongObject" value="6000000" />
                <distribution xsi:type="am:UniformDistribution" />
408           </deviation>
            </value>
410       </default>
        </runnableItems>
412   </runnables>
      <modes name="Message">
414     <literals name="Message_0">
          <customProperties key="enumValue">
416         <value xsi:type="am:LongObject" value="0" />
          </customProperties>
418     </literals>
        <literals name="Message_1">
420       <customProperties key="enumValue">
            <value xsi:type="am:LongObject" value="1" />
```

```
422         </customProperties>
          </literals>
424       <literals name="Message_2">
            <customProperties key="enumValue">
426           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
428       </literals>
          <literals name="Message_3">
430         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="3" />
432         </customProperties>
          </literals>
434       <literals name="Message_4">
            <customProperties key="enumValue">
436           <value xsi:type="am:LongObject" value="4" />
            </customProperties>
438       </literals>
        </modes>
440     <modes name="MessageToT1">
          <literals name="MessageToT1_0">
442         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
444         </customProperties>
          </literals>
446       <literals name="MessageToT1_1">
            <customProperties key="enumValue">
448           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
450       </literals>
        </modes>
452     <modes name="MessageToT2">
          <literals name="MessageToT2_0">
454         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
456         </customProperties>
          </literals>
458       <literals name="MessageToT2_1">
            <customProperties key="enumValue">
460           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
462       </literals>
        </modes>
464     <modes name="StateT1">
          <literals name="StateT1_0">
466         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
468         </customProperties>
          </literals>
470       <literals name="StateT1_1">
            <customProperties key="enumValue">
472           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
474       </literals>
        </modes>
476     <modes name="StateT2">
          <literals name="StateT2_0">
478         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
```

```
480         </customProperties>
          </literals>
482       <literals name="StateT2_1">
            <customProperties key="enumValue">
484           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
486       </literals>
          <literals name="StateT2_2">
488         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
490         </customProperties>
          </literals>
492     </modes>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
494       <size value="8" unit="bit" />
        </modeLabels>
496     <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
          <size value="1" unit="bit" />
498     </modeLabels>
        <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
500       <size value="1" unit="bit" />
        </modeLabels>
502     <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
          <size value="1" unit="bit" />
504     </modeLabels>
        <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
506       <size value="8" unit="bit" />
        </modeLabels>
508   </swModel>
      <hwModel>
510     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
512     </definitions>
        <featureCategories name="Instructions" featureType="performance">
514       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
516     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
518         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
520           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
522             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
524         </structures>
          </structures>
526     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
528       <defaultValue value="600.0" unit="MHz"/>
        </domains>
530   </hwModel>
      <osModel>
532     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
534         <schedulingAlgorithm xsi:type="am:OSEK" />
```

```
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="15" unit="ms" />
      <recurrence value="250" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
      <recurrence value="100" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
      <offset value="15" unit="ms" />
      <recurrence value="60" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
        Runnable_2_Overflow?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_0" entity="Runnable_3_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_1" entity="Runnable_3_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_2" entity="Runnable_3_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_3" entity="Runnable_3_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_4" entity="Runnable_3_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_1" entity="Runnable_4_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_2" entity="Runnable_4_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_3" entity="Runnable_4_3?type=
        Runnable" />
```

```
578     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_4" entity="Runnable_4_4?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_x" entity="Runnable_4_x?type=
            Runnable" />
580     <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
            type=Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
            type=Runnable" />
582     <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
            type=Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
            Runnable_Transition_0?type=Runnable" />
584     <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
            Runnable_Transition_1?type=Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
            Runnable_Transition_2?type=Runnable" />
586     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
            PeriodicStimulus" />
588     <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
            PeriodicStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
            PeriodicStimulus" />
590   </eventModel>
      <mappingModel addressMappingType="offset">
592     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
594     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
596     <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
            stateT2?type=ModeLabel" />
598     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
            stateT1?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
            messageToT1?type=ModeLabel" />
600     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
            message?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
            messageToT2?type=ModeLabel" />
602   </mappingModel>
      <componentsModel />
604 </am:Amalthea>
```

Listing A.31: Variation 2 of State Machine Feedback Loop.

A.1.5.3. Variation 3

```
    <?xml version="1.0" encoding="UTF-8"?>
2   <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
        " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
4       <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
```

```
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                     />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                    StateT1/StateT1_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                     />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                    StateT1/StateT1_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
                MessageToT1/MessageToT1_1?type=ModeLiteral" />
          </condition>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
        </defaultEntry>
      </graphEntries>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_1?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
```

```
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="Stimulus_Task_2?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
```

```
106             <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                      Runnable" />
108               </items>
                  <condition>
110                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
                  </condition>
112             </entries>
              </items>
114           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_0?type=ModeLiteral" />
116           </condition>
            </entries>
118         <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State_2">
120             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
              </items>
122           <items xsi:type="am:ModeSwitch">
                <entries>
124               <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                        Runnable" />
126               </items>
                  <condition>
128                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
                  </condition>
130             </entries>
                <entries>
132               <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
                    <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                        Runnable" />
134               </items>
                  <condition>
136                 <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
                  </condition>
138             </entries>
              </items>
140           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_2?type=ModeLiteral" />
142           </condition>
            </entries>
144       </graphEntries>
        </callGraph>
146     <customProperties key="priority">
          <value xsi:type="am:StringObject" value="1" />
148     </customProperties>
        <customProperties key="osekTaskGroup">
150       <value xsi:type="am:StringObject" value="1" />
        </customProperties>
152   </tasks>
      <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
154     <callGraph>
```

```
      <graphEntries xsi:type="am:ProbabilitySwitch">
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="30.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_2?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_4?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_1?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_0?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_3">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="Stimulus_Task_4?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_3?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_3?type=ModeLiteral" />
```

```
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_1">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_1?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_1?type=ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_4?type=Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_4?type=ModeLiteral" />
            </condition>
          </entries>
          <defaultEntry>
            <items xsi:type="am:CallSequence" name="CallSequence_4_x">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_x?type=Runnable" />
            </items>
          </defaultEntry>
        </graphEntries>
      </callGraph>
      <customProperties key="priority">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
      <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="3" />
      </customProperties>
    </tasks>
    <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:CallSequence" name="CS_Task_5">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
        </graphEntries>
      </callGraph>
    </tasks>
    <runnables name="Runnable_1_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
          " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_State_0" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="59" />
              <upperBound xsi:type="am:LongObject" value="60" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
```

```
264       </runnableItems>
        </runnables>
266     <runnables name="Runnable_State_1" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
268         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
270             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="59400" />
272               <upperBound xsi:type="am:LongObject" value="60000" />
                  <distribution xsi:type="am:UniformDistribution" />
274             </deviation>
              </value>
276         </default>
          </runnableItems>
278     </runnables>
        <runnables name="Runnable_State_2" callback="false" service="false">
280       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
282           <value xsi:type="am:NeedDeviation">
                <deviation>
284               <lowerBound xsi:type="am:LongObject" value="29700000" />
                  <upperBound xsi:type="am:LongObject" value="30000000" />
286               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
288           </value>
            </default>
290       </runnableItems>
        </runnables>
292     <runnables name="Runnable_1" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
294         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
296             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="5940000" />
298               <upperBound xsi:type="am:LongObject" value="6000000" />
                  <distribution xsi:type="am:UniformDistribution" />
300             </deviation>
              </value>
302         </default>
          </runnableItems>
304     </runnables>
        <runnables name="Runnable_1_0" callback="false" service="false">
306       <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
              " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
        </runnables>
308     <runnables name="Runnable_Transition_0" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_0?type=ModeLiteral" />
310     </runnables>
        <runnables name="Runnable_Transition_1" callback="false" service="false">
312       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_1?type=ModeLiteral" />
        </runnables>
314     <runnables name="Runnable_Transition_2" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_2?type=ModeLiteral" />
316     </runnables>
        <runnables name="Runnable_1_State0" callback="false" service="false">
```

```
318       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_0?type=ModeLiteral" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
320     </runnables>
        <runnables name="Runnable_1_State1" callback="false" service="false">
322       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_1?type=ModeLiteral" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
324     </runnables>
        <runnables name="Runnable_2_Overflow" callback="false" service="false">
326       <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
        </runnables>
328     <runnables name="Runnable_3_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_1?type=ModeLiteral" />
330     </runnables>
        <runnables name="Runnable_4_1" callback="false" service="false">
332       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
334           <value xsi:type="am:NeedDeviation">
                <deviation>
336               <lowerBound xsi:type="am:LongObject" value="594" />
                  <upperBound xsi:type="am:LongObject" value="600" />
338               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
340           </value>
            </default>
342       </runnableItems>
        </runnables>
344     <runnables name="Runnable_4_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
346         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
348             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700" />
350               <upperBound xsi:type="am:LongObject" value="30000" />
                  <distribution xsi:type="am:UniformDistribution" />
352             </deviation>
              </value>
354         </default>
          </runnableItems>
356     </runnables>
        <runnables name="Runnable_4_3" callback="false" service="false">
358       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
360           <value xsi:type="am:NeedDeviation">
                <deviation>
362               <lowerBound xsi:type="am:LongObject" value="594000" />
                  <upperBound xsi:type="am:LongObject" value="600000" />
364               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
366           </value>
            </default>
368       </runnableItems>
        </runnables>
```

```
370     <runnables name="Runnable_4_4" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
372         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
374             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="23760000" />
376               <upperBound xsi:type="am:LongObject" value="24000000" />
                  <distribution xsi:type="am:UniformDistribution" />
378             </deviation>
              </value>
380         </default>
          </runnableItems>
382     </runnables>
        <runnables name="Runnable_4_x" callback="false" service="false">
384       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
386           <value xsi:type="am:NeedDeviation">
                <deviation>
388               <lowerBound xsi:type="am:LongObject" value="59" />
                  <upperBound xsi:type="am:LongObject" value="60" />
390               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
392           </value>
            </default>
394       </runnableItems>
        </runnables>
396     <runnables name="Runnable_3_2" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_2?type=ModeLiteral" />
398     </runnables>
        <runnables name="Runnable_3_3" callback="false" service="false">
400       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_3?type=ModeLiteral" />
        </runnables>
402     <runnables name="Runnable_3_4" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_4?type=ModeLiteral" />
404     </runnables>
        <runnables name="Runnable_3_0" callback="false" service="false">
406       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_0?type=ModeLiteral" />
        </runnables>
408     <runnables name="Runnable_3" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
410         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
412             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="5940000" />
414               <upperBound xsi:type="am:LongObject" value="6000000" />
                  <distribution xsi:type="am:UniformDistribution" />
416             </deviation>
              </value>
418         </default>
          </runnableItems>
420     </runnables>
        <runnables name="Runnable_5_1" callback="false" service="false">
422       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
```

```
424           <value xsi:type="am:NeedDeviation">
                <deviation>
426               <lowerBound xsi:type="am:LongObject" value="35640000" />
                  <upperBound xsi:type="am:LongObject" value="36000000" />
428               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
430           </value>
            </default>
432       </runnableItems>
        </runnables>
434     <runnables name="Runnable_5_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
436         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
438             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="11880000" />
440               <upperBound xsi:type="am:LongObject" value="12000000" />
                  <distribution xsi:type="am:UniformDistribution" />
442             </deviation>
              </value>
444         </default>
          </runnableItems>
446     </runnables>
        <modes name="Message">
448       <literals name="Message_0">
            <customProperties key="enumValue">
450           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
452       </literals>
          <literals name="Message_1">
454         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
456         </customProperties>
          </literals>
458       <literals name="Message_2">
            <customProperties key="enumValue">
460           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
462       </literals>
          <literals name="Message_3">
464         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="3" />
466         </customProperties>
          </literals>
468       <literals name="Message_4">
            <customProperties key="enumValue">
470           <value xsi:type="am:LongObject" value="4" />
            </customProperties>
472       </literals>
        </modes>
474     <modes name="MessageToT1">
          <literals name="MessageToT1_0">
476         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
478         </customProperties>
          </literals>
480       <literals name="MessageToT1_1">
            <customProperties key="enumValue">
```

```
482           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
484       </literals>
        </modes>
486     <modes name="MessageToT2">
          <literals name="MessageToT2_0">
488         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
490         </customProperties>
          </literals>
492       <literals name="MessageToT2_1">
            <customProperties key="enumValue">
494           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
496       </literals>
        </modes>
498     <modes name="StateT1">
          <literals name="StateT1_0">
500         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
502         </customProperties>
          </literals>
504       <literals name="StateT1_1">
            <customProperties key="enumValue">
506           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
508       </literals>
        </modes>
510     <modes name="StateT2">
          <literals name="StateT2_0">
512         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
514         </customProperties>
          </literals>
516       <literals name="StateT2_1">
            <customProperties key="enumValue">
518           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
520       </literals>
          <literals name="StateT2_2">
522         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
524         </customProperties>
          </literals>
526     </modes>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
528       <size value="8" unit="bit" />
        </modeLabels>
530     <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
          <size value="1" unit="bit" />
532     </modeLabels>
        <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
534       <size value="1" unit="bit" />
        </modeLabels>
536     <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
          <size value="1" unit="bit" />
538     </modeLabels>
        <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
```

```
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_2">
      <offset value="15" unit="ms" />
      <recurrence value="250" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
      <recurrence value="100" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_4">
      <offset value="15" unit="ms" />
      <recurrence value="60" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="1000" unit="ms" />
    </stimuli>
  </stimuliModel>
```

```
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
        Runnable_2_Overflow?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_0" entity="Runnable_3_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_1" entity="Runnable_3_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_2" entity="Runnable_3_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_3" entity="Runnable_3_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_4" entity="Runnable_3_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_1" entity="Runnable_4_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_2" entity="Runnable_4_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_3" entity="Runnable_4_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_4" entity="Runnable_4_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_x" entity="Runnable_4_x?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
```

```
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_2" entity="Stimulus_Task_2?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_4" entity="Stimulus_Task_4?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
        stateT2?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
        stateT1?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        messageToT1?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
        messageToT2?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.32: Variation 3 of State Machine Feedback Loop.

A.1.5.4. Variation 4

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_State0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                       />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                      StateT1/StateT1_0?type=ModeLiteral" />
                </condition>
```

```
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                    />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                    StateT1/StateT1_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
                MessageToT1/MessageToT1_1?type=ModeLiteral" />
          </condition>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
        </defaultEntry>
      </graphEntries>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_1?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
```

```
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
```

```
116             <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_0?type=ModeLiteral" />
              </condition>
118         </entries>
            <entries>
120           <items xsi:type="am:CallSequence" name="CallSequence_State_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
122           </items>
              <items xsi:type="am:ModeSwitch">
124             <entries>
                  <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
126                 <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                        Runnable" />
                  </items>
128               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
130               </condition>
                </entries>
132             <entries>
                  <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
134                 <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                        Runnable" />
                  </items>
136               <condition>
                    <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                        value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
138               </condition>
                </entries>
140           </items>
              <condition>
142             <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_2?type=ModeLiteral" />
              </condition>
144         </entries>
          </graphEntries>
146     </callGraph>
        <customProperties key="priority">
148       <value xsi:type="am:StringObject" value="1" />
        </customProperties>
150     <customProperties key="osekTaskGroup">
          <value xsi:type="am:StringObject" value="1" />
152     </customProperties>
      </tasks>
154   <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
          multipleTaskActivationLimit="1">
        <callGraph>
156       <graphEntries xsi:type="am:ProbabilitySwitch">
            <entries probability="20.0">
158           <items xsi:type="am:CallSequence" name="CallSequence_3_3">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_3?type=Runnable" />
160           </items>
            </entries>
162         <entries probability="30.0">
              <items xsi:type="am:CallSequence" name="CallSequence_3_2">
164             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_2?type=Runnable" />
              </items>
166         </entries>
```

```
          <entries probability="15.0">
168         <items xsi:type="am:CallSequence" name="CallSequence_3_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_4?type=Runnable" />
170         </items>
          </entries>
172       <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_1">
174           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_1?type=Runnable" />
            </items>
176       </entries>
          <entries probability="15.0">
178         <items xsi:type="am:CallSequence" name="CallSequence_3_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_0?type=Runnable" />
180         </items>
          </entries>
182     </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_3">
184       <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=InterProcessStimulus"
              />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
186     </graphEntries>
      </callGraph>
188   <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
190   </customProperties>
      <customProperties key="osekTaskGroup">
192     <value xsi:type="am:StringObject" value="4" />
      </customProperties>
194 </tasks>
    <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
196   <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
198       <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_2">
200           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_2?type=Runnable" />
            </items>
202         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_2?type=ModeLiteral" />
204         </condition>
          </entries>
206       <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_3">
208           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_3?type=Runnable" />
            </items>
210         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_3?type=ModeLiteral" />
212         </condition>
          </entries>
214       <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_4_1">
216           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_1?type=Runnable" />
            </items>
218         <condition>
              <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                  Message/Message_1?type=ModeLiteral" />
```

```
220           </condition>
            </entries>
222           <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_4_4">
224             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_4?type=Runnable" />
              </items>
226           <condition>
                <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                    Message/Message_4?type=ModeLiteral" />
228           </condition>
            </entries>
230         <defaultEntry>
              <items xsi:type="am:CallSequence" name="CallSequence_4_x">
232             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_x?type=Runnable" />
              </items>
234         </defaultEntry>
          </graphEntries>
236     </callGraph>
      <customProperties key="priority">
238     <value xsi:type="am:StringObject" value="3" />
      </customProperties>
240   <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
242   </customProperties>
  </tasks>
244 <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
246   <graphEntries xsi:type="am:CallSequence" name="CS_Task_5">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
248     <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
      </graphEntries>
250 </callGraph>
  </tasks>
252 <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
        " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
254 </runnables>
  <runnables name="Runnable_State_0" callback="false" service="false">
256   <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
258     <value xsi:type="am:NeedDeviation">
          <deviation>
260         <lowerBound xsi:type="am:LongObject" value="59" />
            <upperBound xsi:type="am:LongObject" value="60" />
262         <distribution xsi:type="am:UniformDistribution" />
          </deviation>
264     </value>
      </default>
266   </runnableItems>
  </runnables>
268 <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
270   <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
272       <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
274         <upperBound xsi:type="am:LongObject" value="60000" />
```

```
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700000" />
            <upperBound xsi:type="am:LongObject" value="30000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6000000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
        " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
        modeValue="StateT2/StateT2_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
        modeValue="StateT2/StateT2_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
        modeValue="StateT2/StateT2_2?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_State0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
        modeValue="StateT1/StateT1_0?type=ModeLiteral" />
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
        " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_1_State1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
        modeValue="StateT1/StateT1_1?type=ModeLiteral" />
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
          " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_2_Overflow" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
          " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_3_1" callback="false" service="false">
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_1?type=ModeLiteral" />
    </runnables>
    <runnables name="Runnable_4_1" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594" />
              <upperBound xsi:type="am:LongObject" value="600" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_4_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="29700" />
              <upperBound xsi:type="am:LongObject" value="30000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_4_3" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="594000" />
              <upperBound xsi:type="am:LongObject" value="600000" />
              <distribution xsi:type="am:UniformDistribution" />
            </deviation>
          </value>
        </default>
      </runnableItems>
    </runnables>
    <runnables name="Runnable_4_4" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
            <deviation>
              <lowerBound xsi:type="am:LongObject" value="23760000" />
              <upperBound xsi:type="am:LongObject" value="24000000" />
              <distribution xsi:type="am:UniformDistribution" />
```

```
380             </deviation>
              </value>
382           </default>
            </runnableItems>
384       </runnables>
          <runnables name="Runnable_4_x" callback="false" service="false">
386         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
388             <value xsi:type="am:NeedDeviation">
                  <deviation>
390                 <lowerBound xsi:type="am:LongObject" value="59" />
                    <upperBound xsi:type="am:LongObject" value="60" />
392                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
394             </value>
              </default>
396         </runnableItems>
          </runnables>
398       <runnables name="Runnable_3_2" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
                modeValue="Message/Message_2?type=ModeLiteral" />
400       </runnables>
          <runnables name="Runnable_3_3" callback="false" service="false">
402         <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
                modeValue="Message/Message_3?type=ModeLiteral" />
          </runnables>
404       <runnables name="Runnable_3_4" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
                modeValue="Message/Message_4?type=ModeLiteral" />
406       </runnables>
          <runnables name="Runnable_3_0" callback="false" service="false">
408         <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
                modeValue="Message/Message_0?type=ModeLiteral" />
          </runnables>
410       <runnables name="Runnable_3" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
412           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
414               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="5940000" />
416                 <upperBound xsi:type="am:LongObject" value="6000000" />
                    <distribution xsi:type="am:UniformDistribution" />
418               </deviation>
                </value>
420           </default>
            </runnableItems>
422       </runnables>
          <runnables name="Runnable_5_1" callback="false" service="false">
424         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
426             <value xsi:type="am:NeedDeviation">
                  <deviation>
428                 <lowerBound xsi:type="am:LongObject" value="35640000" />
                    <upperBound xsi:type="am:LongObject" value="36000000" />
430                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
432             </value>
              </default>
```

```
434       </runnableItems>
        </runnables>
436     <runnables name="Runnable_5_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
438         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
440             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="11880000" />
442               <upperBound xsi:type="am:LongObject" value="12000000" />
                  <distribution xsi:type="am:UniformDistribution" />
444             </deviation>
              </value>
446         </default>
          </runnableItems>
448     </runnables>
        <modes name="Message">
450       <literals name="Message_0">
            <customProperties key="enumValue">
452           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
454       </literals>
          <literals name="Message_1">
456         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
458         </customProperties>
          </literals>
460       <literals name="Message_2">
            <customProperties key="enumValue">
462           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
464       </literals>
          <literals name="Message_3">
466         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="3" />
468         </customProperties>
          </literals>
470       <literals name="Message_4">
            <customProperties key="enumValue">
472           <value xsi:type="am:LongObject" value="4" />
            </customProperties>
474       </literals>
        </modes>
476     <modes name="MessageToT1">
          <literals name="MessageToT1_0">
478         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
480         </customProperties>
          </literals>
482       <literals name="MessageToT1_1">
            <customProperties key="enumValue">
484           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
486       </literals>
        </modes>
488     <modes name="MessageToT2">
          <literals name="MessageToT2_0">
490         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
```

```
492         </customProperties>
          </literals>
494       <literals name="MessageToT2_1">
            <customProperties key="enumValue">
496           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
498       </literals>
        </modes>
500     <modes name="StateT1">
          <literals name="StateT1_0">
502         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
504         </customProperties>
          </literals>
506       <literals name="StateT1_1">
            <customProperties key="enumValue">
508           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
510       </literals>
        </modes>
512     <modes name="StateT2">
          <literals name="StateT2_0">
514         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
516         </customProperties>
          </literals>
518       <literals name="StateT2_1">
            <customProperties key="enumValue">
520           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
522       </literals>
          <literals name="StateT2_2">
524         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
526         </customProperties>
          </literals>
528     </modes>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
530       <size value="8" unit="bit" />
        </modeLabels>
532     <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
          <size value="1" unit="bit" />
534     </modeLabels>
        <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
536       <size value="1" unit="bit" />
        </modeLabels>
538     <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
          <size value="1" unit="bit" />
540     </modeLabels>
        <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
542       <size value="8" unit="bit" />
        </modeLabels>
544   </swModel>
      <hwModel>
546     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
548     </definitions>
```

```
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="300" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
      <recurrence value="100" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="1000" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
        Runnable_2_Overflow?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_0" entity="Runnable_3_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_1" entity="Runnable_3_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_2" entity="Runnable_3_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_3" entity="Runnable_3_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_4" entity="Runnable_3_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_1" entity="Runnable_4_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_2" entity="Runnable_4_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_3" entity="Runnable_4_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_4" entity="Runnable_4_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_x" entity="Runnable_4_x?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" entity="IPA_Task_2?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
```

```
632     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
634     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
636     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
            stateT2?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
            stateT1?type=ModeLabel" />
638     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
            messageToT1?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
            message?type=ModeLabel" />
640     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
            messageToT2?type=ModeLabel" />
    </mappingModel>
642   <componentsModel />
</am:Amalthea>
```

Listing A.33: Variation 4 of State Machine Feedback Loop.

A.1.5.5. Variation 5

```
    <?xml version="1.0" encoding="UTF-8"?>
2   <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
        " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
4       <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
          <callGraph>
6           <graphEntries xsi:type="am:ModeSwitch">
              <entries>
8               <items xsi:type="am:ModeSwitch">
                  <entries>
10                  <items xsi:type="am:CallSequence" name="CallSequence_State0">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                           />
12                  </items>
                    <condition>
14                    <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                           StateT1/StateT1_0?type=ModeLiteral" />
                    </condition>
16                </entries>
                  <entries>
18                  <items xsi:type="am:CallSequence" name="CallSequence_State1">
                      <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                           />
20                  </items>
                    <condition>
22                    <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                           StateT1/StateT1_1?type=ModeLiteral" />
                    </condition>
24                </entries>
                </items>
26              <condition>
```

```
            <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
                MessageToT1/MessageToT1_1?type=ModeLiteral" />
          </condition>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
        </defaultEntry>
      </graphEntries>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_1?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
```

```
78                </condition>
              </entries>
80              <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
82                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                      Runnable" />
                </items>
84              <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
86              </condition>
              </entries>
88          </items>
            <condition>
90            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                  StateT2/StateT2_1?type=ModeLiteral" />
            </condition>
92        </entries>
          <entries>
94          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
96          </items>
            <items xsi:type="am:ModeSwitch">
98            <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
100               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                      Runnable" />
                </items>
102             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
104             </condition>
              </entries>
106           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
108               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                      Runnable" />
                </items>
110             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                      value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
112             </condition>
              </entries>
114         </items>
            <condition>
116           <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                  StateT2/StateT2_0?type=ModeLiteral" />
            </condition>
118       </entries>
          <entries>
120         <items xsi:type="am:CallSequence" name="CallSequence_State_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
122         </items>
            <items xsi:type="am:ModeSwitch">
124           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
126               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                      Runnable" />
```

```
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                  value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
            </condition>
          </entries>
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                  Runnable" />
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                  value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
            </condition>
          </entries>
        </items>
        <condition>
          <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
              StateT2/StateT2_2?type=ModeLiteral" />
        </condition>
      </entries>
    </graphEntries>
  </callGraph>
  <customProperties key="priority">
    <value xsi:type="am:StringObject" value="1" />
  </customProperties>
  <customProperties key="osekTaskGroup">
    <value xsi:type="am:StringObject" value="1" />
  </customProperties>
</tasks>
<tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
    multipleTaskActivationLimit="1">
  <callGraph>
    <graphEntries xsi:type="am:ProbabilitySwitch">
      <entries probability="20.0">
        <items xsi:type="am:CallSequence" name="CallSequence_3_3">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_3?type=Runnable" />
        </items>
      </entries>
      <entries probability="30.0">
        <items xsi:type="am:CallSequence" name="CallSequence_3_2">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_2?type=Runnable" />
        </items>
      </entries>
      <entries probability="15.0">
        <items xsi:type="am:CallSequence" name="CallSequence_3_4">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_4?type=Runnable" />
        </items>
      </entries>
      <entries probability="20.0">
        <items xsi:type="am:CallSequence" name="CallSequence_3_1">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_1?type=Runnable" />
        </items>
      </entries>
      <entries probability="15.0">
        <items xsi:type="am:CallSequence" name="CallSequence_3_0">
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_0?type=Runnable" />
```

```
180           </items>
            </entries>
182         </graphEntries>
            <graphEntries xsi:type="am:CallSequence" name="CallSequence_3">
184           <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=InterProcessStimulus"
                 />
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
186         </graphEntries>
          </callGraph>
188       <customProperties key="priority">
            <value xsi:type="am:StringObject" value="4" />
190       </customProperties>
          <customProperties key="osekTaskGroup">
192         <value xsi:type="am:StringObject" value="4" />
          </customProperties>
194     </tasks>
        <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
            multipleTaskActivationLimit="1">
196       <callGraph>
            <graphEntries xsi:type="am:ModeSwitch">
198           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_2">
200               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_2?type=Runnable" />
                </items>
202             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_2?type=ModeLiteral" />
204             </condition>
              </entries>
206           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_3">
208               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_3?type=Runnable" />
                </items>
210             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_3?type=ModeLiteral" />
212             </condition>
              </entries>
214           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_1">
216               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_1?type=Runnable" />
                </items>
218             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_1?type=ModeLiteral" />
220             </condition>
              </entries>
222           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_4">
224               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_4?type=Runnable" />
                </items>
226             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_4?type=ModeLiteral" />
228             </condition>
              </entries>
230           <defaultEntry>
                <items xsi:type="am:CallSequence" name="CallSequence_4_x">
```

```
232             <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_x?type=Runnable" />
              </items>
234           </defaultEntry>
          </graphEntries>
236       </callGraph>
      <customProperties key="priority">
238       <value xsi:type="am:StringObject" value="3" />
      </customProperties>
240   <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
242   </customProperties>
  </tasks>
244 <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
246   <graphEntries xsi:type="am:CallSequence" name="CS_Task_5">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
248     <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
      </graphEntries>
250 </callGraph>
  </tasks>
252 <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
        " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
254 </runnables>
  <runnables name="Runnable_State_0" callback="false" service="false">
256   <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
258     <value xsi:type="am:NeedDeviation">
        <deviation>
260       <lowerBound xsi:type="am:LongObject" value="59" />
          <upperBound xsi:type="am:LongObject" value="60" />
262       <distribution xsi:type="am:UniformDistribution" />
        </deviation>
264     </value>
      </default>
266   </runnableItems>
  </runnables>
268 <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
270   <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
272       <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
274         <upperBound xsi:type="am:LongObject" value="60000" />
            <distribution xsi:type="am:UniformDistribution" />
276       </deviation>
        </value>
278   </default>
    </runnableItems>
280 </runnables>
  <runnables name="Runnable_State_2" callback="false" service="false">
282   <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
284     <value xsi:type="am:NeedDeviation">
        <deviation>
286       <lowerBound xsi:type="am:LongObject" value="29700000" />
          <upperBound xsi:type="am:LongObject" value="30000000" />
```

```
288             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
290           </value>
            </default>
292       </runnableItems>
        </runnables>
294     <runnables name="Runnable_1" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
296         <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
298           <deviation>
              <lowerBound xsi:type="am:LongObject" value="5940000" />
300           <upperBound xsi:type="am:LongObject" value="6000000" />
              <distribution xsi:type="am:UniformDistribution" />
302           </deviation>
            </value>
304         </default>
          </runnableItems>
306     </runnables>
        <runnables name="Runnable_1_0" callback="false" service="false">
308       <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
              " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
        </runnables>
310     <runnables name="Runnable_Transition_0" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_0?type=ModeLiteral" />
312     </runnables>
        <runnables name="Runnable_Transition_1" callback="false" service="false">
314       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_1?type=ModeLiteral" />
        </runnables>
316     <runnables name="Runnable_Transition_2" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_2?type=ModeLiteral" />
318     </runnables>
        <runnables name="Runnable_1_State0" callback="false" service="false">
320       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_0?type=ModeLiteral" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
322     </runnables>
        <runnables name="Runnable_1_State1" callback="false" service="false">
324       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_1?type=ModeLiteral" />
          <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
326     </runnables>
        <runnables name="Runnable_2_Overflow" callback="false" service="false">
328       <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
        </runnables>
330     <runnables name="Runnable_3_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_1?type=ModeLiteral" />
332     </runnables>
        <runnables name="Runnable_4_1" callback="false" service="false">
334       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
```

```
336           <value xsi:type="am:NeedDeviation">
                <deviation>
338               <lowerBound xsi:type="am:LongObject" value="594" />
                  <upperBound xsi:type="am:LongObject" value="600" />
340               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
342           </value>
            </default>
344       </runnableItems>
        </runnables>
346     <runnables name="Runnable_4_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
348         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
350             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700" />
352               <upperBound xsi:type="am:LongObject" value="30000" />
                  <distribution xsi:type="am:UniformDistribution" />
354             </deviation>
              </value>
356         </default>
          </runnableItems>
358     </runnables>
        <runnables name="Runnable_4_3" callback="false" service="false">
360       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
362           <value xsi:type="am:NeedDeviation">
                <deviation>
364               <lowerBound xsi:type="am:LongObject" value="594000" />
                  <upperBound xsi:type="am:LongObject" value="600000" />
366               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
368           </value>
            </default>
370       </runnableItems>
        </runnables>
372     <runnables name="Runnable_4_4" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
374         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
376             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="23760000" />
378               <upperBound xsi:type="am:LongObject" value="24000000" />
                  <distribution xsi:type="am:UniformDistribution" />
380             </deviation>
              </value>
382         </default>
          </runnableItems>
384     </runnables>
        <runnables name="Runnable_4_x" callback="false" service="false">
386       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
388           <value xsi:type="am:NeedDeviation">
                <deviation>
390               <lowerBound xsi:type="am:LongObject" value="59" />
                  <upperBound xsi:type="am:LongObject" value="60" />
392               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
```

```
394           </value>
            </default>
396       </runnableItems>
        </runnables>
398     <runnables name="Runnable_3_2" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_2?type=ModeLiteral" />
400     </runnables>
        <runnables name="Runnable_3_3" callback="false" service="false">
402       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_3?type=ModeLiteral" />
        </runnables>
404     <runnables name="Runnable_3_4" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_4?type=ModeLiteral" />
406     </runnables>
        <runnables name="Runnable_3_0" callback="false" service="false">
408       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_0?type=ModeLiteral" />
        </runnables>
410     <runnables name="Runnable_3" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
412         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
414             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="5940000" />
416               <upperBound xsi:type="am:LongObject" value="6000000" />
                  <distribution xsi:type="am:UniformDistribution" />
418             </deviation>
              </value>
420         </default>
          </runnableItems>
422     </runnables>
        <runnables name="Runnable_5_1" callback="false" service="false">
424       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
426           <value xsi:type="am:NeedDeviation">
                <deviation>
428               <lowerBound xsi:type="am:LongObject" value="35640000" />
                  <upperBound xsi:type="am:LongObject" value="36000000" />
430               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
432           </value>
            </default>
434       </runnableItems>
        </runnables>
436     <runnables name="Runnable_5_2" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
438         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
440             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="11880000" />
442               <upperBound xsi:type="am:LongObject" value="12000000" />
                  <distribution xsi:type="am:UniformDistribution" />
444             </deviation>
              </value>
446         </default>
          </runnableItems>
```

```
448     </runnables>
        <modes name="Message">
450       <literals name="Message_0">
            <customProperties key="enumValue">
452           <value xsi:type="am:LongObject" value="0" />
            </customProperties>
454       </literals>
          <literals name="Message_1">
456         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="1" />
458         </customProperties>
          </literals>
460       <literals name="Message_2">
            <customProperties key="enumValue">
462           <value xsi:type="am:LongObject" value="2" />
            </customProperties>
464       </literals>
          <literals name="Message_3">
466         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="3" />
468         </customProperties>
          </literals>
470       <literals name="Message_4">
            <customProperties key="enumValue">
472           <value xsi:type="am:LongObject" value="4" />
            </customProperties>
474       </literals>
        </modes>
476     <modes name="MessageToT1">
          <literals name="MessageToT1_0">
478         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
480         </customProperties>
          </literals>
482       <literals name="MessageToT1_1">
            <customProperties key="enumValue">
484           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
486       </literals>
        </modes>
488     <modes name="MessageToT2">
          <literals name="MessageToT2_0">
490         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
492         </customProperties>
          </literals>
494       <literals name="MessageToT2_1">
            <customProperties key="enumValue">
496           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
498       </literals>
        </modes>
500     <modes name="StateT1">
          <literals name="StateT1_0">
502         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
504         </customProperties>
          </literals>
```

```
506       <literals name="StateT1_1">
            <customProperties key="enumValue">
508           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
510       </literals>
        </modes>
512     <modes name="StateT2">
          <literals name="StateT2_0">
514         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
516         </customProperties>
          </literals>
518       <literals name="StateT2_1">
            <customProperties key="enumValue">
520           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
522       </literals>
          <literals name="StateT2_2">
524         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
526         </customProperties>
          </literals>
528     </modes>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
530       <size value="8" unit="bit" />
        </modeLabels>
532     <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
          <size value="1" unit="bit" />
534     </modeLabels>
        <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
536       <size value="1" unit="bit" />
        </modeLabels>
538     <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
          <size value="1" unit="bit" />
540     </modeLabels>
        <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
542       <size value="8" unit="bit" />
        </modeLabels>
544   </swModel>
      <hwModel>
546     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
548     </definitions>
        <featureCategories name="Instructions" featureType="performance">
550       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
552     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
554         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
556           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
558             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
560         </structures>
```

```
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
  </osModel>
  <stimuliModel>
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
      <offset value="0" unit="ms" />
      <recurrence value="220" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
      <offset value="0" unit="ms" />
      <recurrence value="50" unit="ms" />
    </stimuli>
    <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
    <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
      <offset value="0" unit="ms" />
      <recurrence value="500" unit="ms" />
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
        Runnable_2_Overflow?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_0" entity="Runnable_3_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_1" entity="Runnable_3_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_2" entity="Runnable_3_2?type=
        Runnable" />
```

```
608     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_3" entity="Runnable_3_3?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_4" entity="Runnable_3_4?type=
            Runnable" />
610     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_1" entity="Runnable_4_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_2" entity="Runnable_4_2?type=
            Runnable" />
612     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_3" entity="Runnable_4_3?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_4" entity="Runnable_4_4?type=
            Runnable" />
614     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_x" entity="Runnable_4_x?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
            Runnable" />
616     <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
            type=Runnable" />
618     <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
            type=Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
            type=Runnable" />
620     <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
            Runnable_Transition_0?type=Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
            Runnable_Transition_1?type=Runnable" />
622     <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
            Runnable_Transition_2?type=Runnable" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
            PeriodicStimulus" />
624     <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" entity="IPA_Task_2?type=
            InterProcessStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
            PeriodicStimulus" />
626     <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
            InterProcessStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
            PeriodicStimulus" />
628   </eventModel>
      <mappingModel addressMappingType="offset">
630     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
632     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
634     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
636     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
            stateT2?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
            stateT1?type=ModeLabel" />
638     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
            messageToT1?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
            message?type=ModeLabel" />
```

```
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
        messageToT2?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.34: Variation 5 of State Machine Feedback Loop.

A.1.5.6. Variation 6

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
  <swModel>
    <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:ModeSwitch">
              <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_State0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                      />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                      StateT1/StateT1_0?type=ModeLiteral" />
                </condition>
              </entries>
              <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_State1">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                      />
                </items>
                <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                      StateT1/StateT1_1?type=ModeLiteral" />
                </condition>
              </entries>
            </items>
            <condition>
              <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
                  MessageToT1/MessageToT1_1?type=ModeLiteral" />
            </condition>
          </entries>
          <defaultEntry>
            <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
          </defaultEntry>
        </graphEntries>
        <graphEntries xsi:type="am:ModeSwitch">
          <entries>
            <items xsi:type="am:CallSequence" name="CallSequence_1_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
            </items>
            <condition>
```

```
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_1?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
```

```
90              <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_1?type=ModeLiteral" />
                </condition>
92            </entries>
              <entries>
94              <items xsi:type="am:CallSequence" name="CallSequence_State_0">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
96              </items>
                <items xsi:type="am:ModeSwitch">
98                <entries>
                    <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
100                   <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                          Runnable" />
                    </items>
102                 <condition>
                      <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                          value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
104                 </condition>
                  </entries>
106               <entries>
                    <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
108                   <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                          Runnable" />
                    </items>
110                 <condition>
                      <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                          value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
112                 </condition>
                  </entries>
114             </items>
                <condition>
116               <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                      StateT2/StateT2_0?type=ModeLiteral" />
                </condition>
118           </entries>
              <entries>
120             <items xsi:type="am:CallSequence" name="CallSequence_State_2">
                  <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
122             </items>
                <items xsi:type="am:ModeSwitch">
124               <entries>
                    <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
126                   <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                          Runnable" />
                    </items>
128                 <condition>
                      <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                          value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
130                 </condition>
                  </entries>
132               <entries>
                    <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
134                   <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                          Runnable" />
                    </items>
136                 <condition>
                      <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                          value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
```

```
138               </condition>
              </entries>
140           </items>
              <condition>
142             <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                    StateT2/StateT2_2?type=ModeLiteral" />
              </condition>
144         </entries>
        </graphEntries>
146   </callGraph>
      <customProperties key="priority">
148     <value xsi:type="am:StringObject" value="1" />
      </customProperties>
150   <customProperties key="osekTaskGroup">
        <value xsi:type="am:StringObject" value="1" />
152   </customProperties>
    </tasks>
154 <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
        multipleTaskActivationLimit="1">
      <callGraph>
156     <graphEntries xsi:type="am:ProbabilitySwitch">
          <entries probability="20.0">
158         <items xsi:type="am:CallSequence" name="CallSequence_3_3">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_3?type=Runnable" />
160         </items>
          </entries>
162       <entries probability="30.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_2">
164           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_2?type=Runnable" />
            </items>
166       </entries>
          <entries probability="15.0">
168         <items xsi:type="am:CallSequence" name="CallSequence_3_4">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_4?type=Runnable" />
170         </items>
          </entries>
172       <entries probability="20.0">
            <items xsi:type="am:CallSequence" name="CallSequence_3_1">
174           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_1?type=Runnable" />
            </items>
176       </entries>
          <entries probability="15.0">
178         <items xsi:type="am:CallSequence" name="CallSequence_3_0">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_0?type=Runnable" />
180         </items>
          </entries>
182     </graphEntries>
        <graphEntries xsi:type="am:CallSequence" name="CallSequence_3">
184       <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=InterProcessStimulus"
              />
          <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
186     </graphEntries>
      </callGraph>
188   <customProperties key="priority">
        <value xsi:type="am:StringObject" value="4" />
190   </customProperties>
      <customProperties key="osekTaskGroup">
192     <value xsi:type="am:StringObject" value="4" />
```

```
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_3?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_3?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_4?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_4?type=ModeLiteral" />
          </condition>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_4_x">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_x?type=Runnable" />
          </items>
        </defaultEntry>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="3" />
    </customProperties>
  </tasks>
  <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="1">
```

```
    <callGraph>
      <graphEntries xsi:type="am:CallSequence" name="CS_Task_5">
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
      </graphEntries>
    </callGraph>
  </tasks>
  <runnables name="Runnable_1_1" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
        " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_State_0" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="58" />
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
            <upperBound xsi:type="am:LongObject" value="60600" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700000" />
            <upperBound xsi:type="am:LongObject" value="30300000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
```

```
302             </deviation>
              </value>
304           </default>
        </runnableItems>
306   </runnables>
      <runnables name="Runnable_1_0" callback="false" service="false">
308     <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
            " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
      </runnables>
310   <runnables name="Runnable_Transition_0" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
            modeValue="StateT2/StateT2_0?type=ModeLiteral" />
312   </runnables>
      <runnables name="Runnable_Transition_1" callback="false" service="false">
314     <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
            modeValue="StateT2/StateT2_1?type=ModeLiteral" />
      </runnables>
316   <runnables name="Runnable_Transition_2" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
            modeValue="StateT2/StateT2_2?type=ModeLiteral" />
318   </runnables>
      <runnables name="Runnable_1_State0" callback="false" service="false">
320     <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
            modeValue="StateT1/StateT1_0?type=ModeLiteral" />
        <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
            " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
322   </runnables>
      <runnables name="Runnable_1_State1" callback="false" service="false">
324     <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
            modeValue="StateT1/StateT1_1?type=ModeLiteral" />
        <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
            " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
326   </runnables>
      <runnables name="Runnable_2_Overflow" callback="false" service="false">
328     <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
            " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
      </runnables>
330   <runnables name="Runnable_3_1" callback="false" service="false">
        <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
            modeValue="Message/Message_1?type=ModeLiteral" />
332   </runnables>
      <runnables name="Runnable_4_1" callback="false" service="false">
334     <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
336         <value xsi:type="am:NeedDeviation">
              <deviation>
338             <lowerBound xsi:type="am:LongObject" value="594" />
                <upperBound xsi:type="am:LongObject" value="606" />
340             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
342         </value>
          </default>
344     </runnableItems>
      </runnables>
346   <runnables name="Runnable_4_2" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
348       <default key="Instructions">
            <value xsi:type="am:NeedDeviation">
```

```
350             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="29700" />
352               <upperBound xsi:type="am:LongObject" value="30300" />
                  <distribution xsi:type="am:UniformDistribution" />
354             </deviation>
              </value>
356         </default>
          </runnableItems>
358     </runnables>
        <runnables name="Runnable_4_3" callback="false" service="false">
360       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
362           <value xsi:type="am:NeedDeviation">
                <deviation>
364               <lowerBound xsi:type="am:LongObject" value="594000" />
                  <upperBound xsi:type="am:LongObject" value="606000" />
366               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
368           </value>
            </default>
370       </runnableItems>
        </runnables>
372     <runnables name="Runnable_4_4" callback="false" service="false">
          <runnableItems xsi:type="am:ExecutionNeed">
374         <default key="Instructions">
              <value xsi:type="am:NeedDeviation">
376             <deviation>
                  <lowerBound xsi:type="am:LongObject" value="23760000" />
378               <upperBound xsi:type="am:LongObject" value="24240000" />
                  <distribution xsi:type="am:UniformDistribution" />
380             </deviation>
              </value>
382         </default>
          </runnableItems>
384     </runnables>
        <runnables name="Runnable_4_x" callback="false" service="false">
386       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
388           <value xsi:type="am:NeedDeviation">
                <deviation>
390               <lowerBound xsi:type="am:LongObject" value="58" />
                  <upperBound xsi:type="am:LongObject" value="60" />
392               <distribution xsi:type="am:UniformDistribution" />
                </deviation>
394           </value>
            </default>
396       </runnableItems>
        </runnables>
398     <runnables name="Runnable_3_2" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_2?type=ModeLiteral" />
400     </runnables>
        <runnables name="Runnable_3_3" callback="false" service="false">
402       <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_3?type=ModeLiteral" />
        </runnables>
404     <runnables name="Runnable_3_4" callback="false" service="false">
```

```
      <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_4?type=ModeLiteral" />
406   </runnables>
      <runnables name="Runnable_3_0" callback="false" service="false">
408     <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
          modeValue="Message/Message_0?type=ModeLiteral" />
      </runnables>
410   <runnables name="Runnable_3" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
412       <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
414         <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
416         <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
418       </deviation>
          </value>
420     </default>
        </runnableItems>
422   </runnables>
      <runnables name="Runnable_5_1" callback="false" service="false">
424     <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
426       <value xsi:type="am:NeedDeviation">
          <deviation>
428         <lowerBound xsi:type="am:LongObject" value="35640000" />
            <upperBound xsi:type="am:LongObject" value="36360000" />
430         <distribution xsi:type="am:UniformDistribution" />
          </deviation>
432       </value>
        </default>
434     </runnableItems>
      </runnables>
436   <runnables name="Runnable_5_2" callback="false" service="false">
        <runnableItems xsi:type="am:ExecutionNeed">
438       <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
440         <deviation>
            <lowerBound xsi:type="am:LongObject" value="11880000" />
442         <upperBound xsi:type="am:LongObject" value="12120000" />
            <distribution xsi:type="am:UniformDistribution" />
444       </deviation>
          </value>
446     </default>
        </runnableItems>
448   </runnables>
      <modes name="Message">
450     <literals name="Message_0">
        <customProperties key="enumValue">
452       <value xsi:type="am:LongObject" value="0" />
        </customProperties>
454     </literals>
        <literals name="Message_1">
456       <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
458       </customProperties>
        </literals>
460     <literals name="Message_2">
```

```
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="2" />
      </customProperties>
    </literals>
    <literals name="Message_3">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="3" />
      </customProperties>
    </literals>
    <literals name="Message_4">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="4" />
      </customProperties>
    </literals>
  </modes>
  <modes name="MessageToT1">
    <literals name="MessageToT1_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="MessageToT1_1">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="1" />
      </customProperties>
    </literals>
  </modes>
  <modes name="MessageToT2">
    <literals name="MessageToT2_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="MessageToT2_1">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="1" />
      </customProperties>
    </literals>
  </modes>
  <modes name="StateT1">
    <literals name="StateT1_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="StateT1_1">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="1" />
      </customProperties>
    </literals>
  </modes>
  <modes name="StateT2">
    <literals name="StateT2_0">
      <customProperties key="enumValue">
        <value xsi:type="am:LongObject" value="0" />
      </customProperties>
    </literals>
    <literals name="StateT2_1">
```

```
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
        </customProperties>
      </literals>
      <literals name="StateT2_2">
        <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="2" />
        </customProperties>
      </literals>
    </modes>
    <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
    <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
      <size value="1" unit="bit" />
    </modeLabels>
    <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
      <size value="8" unit="bit" />
    </modeLabels>
  </swModel>
  <hwModel>
    <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
        IPC_1.0?type=HwFeature" puType="CPU"/>
    <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
    </definitions>
    <featureCategories name="Instructions" featureType="performance">
      <features name="IPC_1.0" value="1.0" />
    </featureCategories>
    <structures name="System" structureType="System">
      <structures name="Ecu_1" structureType="ECU">
        <structures name="Processor_1" structureType="Microcontroller">
          <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
          </modules>
          <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
              FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
            <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
          </modules>
        </structures>
      </structures>
    </structures>
    <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
      <defaultValue value="600.0" unit="MHz"/>
    </domains>
  </hwModel>
  <osModel>
    <operatingSystems name="Generic_OS">
      <taskSchedulers name="Scheduler_1">
        <schedulingAlgorithm xsi:type="am:OSEK" />
      </taskSchedulers>
      <osDataConsistency mode="noProtection" />
    </operatingSystems>
```

```
574   </osModel>
      <stimuliModel>
576     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
          <offset value="0" unit="ms" />
578       <recurrence value="220" unit="ms" />
        </stimuli>
580     <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
        <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
582       <offset value="0" unit="ms" />
          <recurrence value="50" unit="ms" />
584     </stimuli>
        <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
586     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
          <offset value="0" unit="ms" />
588       <recurrence value="500" unit="ms" />
        </stimuli>
590   </stimuliModel>
      <constraintsModel />
592   <eventModel>
        <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
594     <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
        <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
596     <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
        <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
598     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
            />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
            Runnable" />
600     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
            type=Runnable" />
602     <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
            type=Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
            Runnable_2_Overflow?type=Runnable" />
604     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
            />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_0" entity="Runnable_3_0?type=
            Runnable" />
606     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_1" entity="Runnable_3_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_2" entity="Runnable_3_2?type=
            Runnable" />
608     <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_3" entity="Runnable_3_3?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_4" entity="Runnable_3_4?type=
            Runnable" />
610     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_1" entity="Runnable_4_1?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_2" entity="Runnable_4_2?type=
            Runnable" />
612     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_3" entity="Runnable_4_3?type=
            Runnable" />
        <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_4" entity="Runnable_4_4?type=
            Runnable" />
614     <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_x" entity="Runnable_4_x?type=
            Runnable" />
```

```
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
        Runnable_Transition_2?type=Runnable" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" entity="IPA_Task_2?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
        PeriodicStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
        InterProcessStimulus" />
    <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
        PeriodicStimulus" />
  </eventModel>
  <mappingModel addressMappingType="offset">
    <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
    <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
        ProcessingUnit" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
        stateT2?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
        stateT1?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
        messageToT1?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
        message?type=ModeLabel" />
    <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
        messageToT2?type=ModeLabel" />
  </mappingModel>
  <componentsModel />
</am:Amalthea>
```

Listing A.35: Variation 6 of State Machine Feedback Loop.

A.1.5.7. Variation 7

```
<?xml version="1.0" encoding="UTF-8"?>
<am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.1" xmlns:xmi="http://www.omg.org/XMI
    " xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
```

```
<swModel>
  <tasks name="Task_1" stimuli="Stimulus_Task_1?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State1?type=Runnable"
                    />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                    StateT1/StateT1_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_State1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_State0?type=Runnable"
                    />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                    StateT1/StateT1_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="messageToT1?type=ModeLabel" value="
                MessageToT1/MessageToT1_1?type=ModeLiteral" />
          </condition>
        </entries>
        <defaultEntry>
          <items xsi:type="am:CallSequence" name="CallSequence_Nothing" />
        </defaultEntry>
      </graphEntries>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_0?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_1_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1_1?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT1?type=ModeLabel" value="
                StateT1/StateT1_1?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_1">
```

```
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_2?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_1?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="2" />
    </customProperties>
  </tasks>
  <tasks name="Task_2" stimuli="IPA_Task_2?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_1?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_0?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_1_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_1?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_0?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_0_0">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                    Runnable" />
              </items>
              <condition>
```

```
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_0_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_2?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_0?type=ModeLiteral" />
          </condition>
        </entries>
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_State_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_State_2?type=Runnable" />
          </items>
          <items xsi:type="am:ModeSwitch">
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_2_1">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_Transition_1?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_0?type=ModeLiteral" />
              </condition>
            </entries>
            <entries>
              <items xsi:type="am:CallSequence" name="CallSequence_2_2_2">
                <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_2_Overflow?type=
                    Runnable" />
              </items>
              <condition>
                <entries xsi:type="am:ModeValue" valueProvider="messageToT2?type=ModeLabel"
                    value="MessageToT2/MessageToT2_1?type=ModeLiteral" />
              </condition>
            </entries>
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="stateT2?type=ModeLabel" value="
                StateT2/StateT2_2?type=ModeLiteral" />
          </condition>
        </entries>
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="1" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="1" />
```

```
    </customProperties>
  </tasks>
  <tasks name="Task_3" stimuli="Stimulus_Task_3?type=PeriodicStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ProbabilitySwitch">
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_3">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_3?type=Runnable" />
          </items>
        </entries>
        <entries probability="30.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_2?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_4">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_4?type=Runnable" />
          </items>
        </entries>
        <entries probability="20.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_1">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_1?type=Runnable" />
          </items>
        </entries>
        <entries probability="15.0">
          <items xsi:type="am:CallSequence" name="CallSequence_3_0">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3_0?type=Runnable" />
          </items>
        </entries>
      </graphEntries>
      <graphEntries xsi:type="am:CallSequence" name="CallSequence_3">
        <calls xsi:type="am:InterProcessTrigger" stimulus="IPA_Task_4?type=InterProcessStimulus"
            />
        <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_3?type=Runnable" />
      </graphEntries>
    </callGraph>
    <customProperties key="priority">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
    <customProperties key="osekTaskGroup">
      <value xsi:type="am:StringObject" value="4" />
    </customProperties>
  </tasks>
  <tasks name="Task_4" stimuli="IPA_Task_4?type=InterProcessStimulus" preemption="preemptive"
      multipleTaskActivationLimit="2">
    <callGraph>
      <graphEntries xsi:type="am:ModeSwitch">
        <entries>
          <items xsi:type="am:CallSequence" name="CallSequence_4_2">
            <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_2?type=Runnable" />
          </items>
          <condition>
            <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                Message/Message_2?type=ModeLiteral" />
          </condition>
        </entries>
```

```
206           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_3">
208               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_3?type=Runnable" />
                </items>
210             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_3?type=ModeLiteral" />
212             </condition>
              </entries>
214           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_1">
216               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_1?type=Runnable" />
                </items>
218             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_1?type=ModeLiteral" />
220             </condition>
              </entries>
222           <entries>
                <items xsi:type="am:CallSequence" name="CallSequence_4_4">
224               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_4?type=Runnable" />
                </items>
226             <condition>
                  <entries xsi:type="am:ModeValue" valueProvider="message?type=ModeLabel" value="
                      Message/Message_4?type=ModeLiteral" />
228             </condition>
              </entries>
230           <defaultEntry>
                <items xsi:type="am:CallSequence" name="CallSequence_4_x">
232               <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_4_x?type=Runnable" />
                </items>
234           </defaultEntry>
            </graphEntries>
236       </callGraph>
          <customProperties key="priority">
238         <value xsi:type="am:StringObject" value="3" />
          </customProperties>
240       <customProperties key="osekTaskGroup">
            <value xsi:type="am:StringObject" value="3" />
242       </customProperties>
        </tasks>
244     <tasks name="Task_5" stimuli="Stimulus_Task_5?type=PeriodicStimulus" preemption="preemptive"
            multipleTaskActivationLimit="2">
          <callGraph>
246         <graphEntries xsi:type="am:CallSequence" name="CS_Task_5">
              <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_1?type=Runnable" />
248           <calls xsi:type="am:TaskRunnableCall" runnable="Runnable_5_2?type=Runnable" />
            </graphEntries>
250       </callGraph>
        </tasks>
252     <runnables name="Runnable_1_1" callback="false" service="false">
          <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
              " modeValue="MessageToT2/MessageToT2_1?type=ModeLiteral" />
254     </runnables>
        <runnables name="Runnable_State_0" callback="false" service="false">
256       <runnableItems xsi:type="am:ExecutionNeed">
            <default key="Instructions">
258           <value xsi:type="am:NeedDeviation">
```

```
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="58" />
            <upperBound xsi:type="am:LongObject" value="60" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="59400" />
            <upperBound xsi:type="am:LongObject" value="60600" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_State_2" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="29700000" />
            <upperBound xsi:type="am:LongObject" value="30300000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
      <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
          <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
            <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
          </deviation>
        </value>
      </default>
    </runnableItems>
  </runnables>
  <runnables name="Runnable_1_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT2?type=ModeLabel" access="write
        " modeValue="MessageToT2/MessageToT2_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_0" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
        modeValue="StateT2/StateT2_0?type=ModeLiteral" />
  </runnables>
  <runnables name="Runnable_Transition_1" callback="false" service="false">
```

```
314       <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_1?type=ModeLiteral" />
          </runnables>
316       <runnables name="Runnable_Transition_2" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="stateT2?type=ModeLabel" access="write"
              modeValue="StateT2/StateT2_2?type=ModeLiteral" />
318       </runnables>
          <runnables name="Runnable_1_State0" callback="false" service="false">
320         <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_0?type=ModeLiteral" />
            <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
322       </runnables>
          <runnables name="Runnable_1_State1" callback="false" service="false">
324         <runnableItems xsi:type="am:ModeLabelAccess" data="stateT1?type=ModeLabel" access="write"
              modeValue="StateT1/StateT1_1?type=ModeLiteral" />
            <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_0?type=ModeLiteral" />
326       </runnables>
          <runnables name="Runnable_2_Overflow" callback="false" service="false">
328         <runnableItems xsi:type="am:ModeLabelAccess" data="messageToT1?type=ModeLabel" access="write
              " modeValue="MessageToT1/MessageToT1_1?type=ModeLiteral" />
          </runnables>
330       <runnables name="Runnable_3_1" callback="false" service="false">
            <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
              modeValue="Message/Message_1?type=ModeLiteral" />
332       </runnables>
          <runnables name="Runnable_4_1" callback="false" service="false">
334         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
336             <value xsi:type="am:NeedDeviation">
                  <deviation>
338                 <lowerBound xsi:type="am:LongObject" value="594" />
                    <upperBound xsi:type="am:LongObject" value="606" />
340                 <distribution xsi:type="am:UniformDistribution" />
                  </deviation>
342             </value>
              </default>
344         </runnableItems>
          </runnables>
346       <runnables name="Runnable_4_2" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
348           <default key="Instructions">
                <value xsi:type="am:NeedDeviation">
350               <deviation>
                    <lowerBound xsi:type="am:LongObject" value="29700" />
352                 <upperBound xsi:type="am:LongObject" value="30300" />
                    <distribution xsi:type="am:UniformDistribution" />
354               </deviation>
                </value>
356           </default>
            </runnableItems>
358       </runnables>
          <runnables name="Runnable_4_3" callback="false" service="false">
360         <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
362             <value xsi:type="am:NeedDeviation">
                  <deviation>
```

```
364             <lowerBound xsi:type="am:LongObject" value="594000" />
                <upperBound xsi:type="am:LongObject" value="606000" />
366             <distribution xsi:type="am:UniformDistribution" />
              </deviation>
368         </value>
          </default>
370     </runnableItems>
    </runnables>
372 <runnables name="Runnable_4_4" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
374   <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
376       <deviation>
            <lowerBound xsi:type="am:LongObject" value="23760000" />
378         <upperBound xsi:type="am:LongObject" value="24240000" />
            <distribution xsi:type="am:UniformDistribution" />
380       </deviation>
        </value>
382   </default>
    </runnableItems>
384 </runnables>
    <runnables name="Runnable_4_x" callback="false" service="false">
386   <runnableItems xsi:type="am:ExecutionNeed">
        <default key="Instructions">
388       <value xsi:type="am:NeedDeviation">
            <deviation>
390           <lowerBound xsi:type="am:LongObject" value="58" />
              <upperBound xsi:type="am:LongObject" value="60" />
392           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
394       </value>
        </default>
396   </runnableItems>
    </runnables>
398 <runnables name="Runnable_3_2" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_2?type=ModeLiteral" />
400 </runnables>
    <runnables name="Runnable_3_3" callback="false" service="false">
402   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_3?type=ModeLiteral" />
    </runnables>
404 <runnables name="Runnable_3_4" callback="false" service="false">
    <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_4?type=ModeLiteral" />
406 </runnables>
    <runnables name="Runnable_3_0" callback="false" service="false">
408   <runnableItems xsi:type="am:ModeLabelAccess" data="message?type=ModeLabel" access="write"
        modeValue="Message/Message_0?type=ModeLiteral" />
    </runnables>
410 <runnables name="Runnable_3" callback="false" service="false">
    <runnableItems xsi:type="am:ExecutionNeed">
412   <default key="Instructions">
        <value xsi:type="am:NeedDeviation">
414       <deviation>
            <lowerBound xsi:type="am:LongObject" value="5940000" />
416         <upperBound xsi:type="am:LongObject" value="6060000" />
            <distribution xsi:type="am:UniformDistribution" />
```

```
418             </deviation>
            </value>
420         </default>
        </runnableItems>
422     </runnables>
    <runnables name="Runnable_5_1" callback="false" service="false">
424       <runnableItems xsi:type="am:ExecutionNeed">
          <default key="Instructions">
426         <value xsi:type="am:NeedDeviation">
            <deviation>
428           <lowerBound xsi:type="am:LongObject" value="35640000" />
              <upperBound xsi:type="am:LongObject" value="36360000" />
430           <distribution xsi:type="am:UniformDistribution" />
            </deviation>
432         </value>
          </default>
434       </runnableItems>
    </runnables>
436 <runnables name="Runnable_5_2" callback="false" service="false">
      <runnableItems xsi:type="am:ExecutionNeed">
438     <default key="Instructions">
          <value xsi:type="am:NeedDeviation">
440         <deviation>
              <lowerBound xsi:type="am:LongObject" value="11880000" />
442           <upperBound xsi:type="am:LongObject" value="12120000" />
              <distribution xsi:type="am:UniformDistribution" />
444         </deviation>
          </value>
446     </default>
      </runnableItems>
448 </runnables>
    <modes name="Message">
450   <literals name="Message_0">
        <customProperties key="enumValue">
452       <value xsi:type="am:LongObject" value="0" />
        </customProperties>
454   </literals>
      <literals name="Message_1">
456     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="1" />
458     </customProperties>
      </literals>
460   <literals name="Message_2">
        <customProperties key="enumValue">
462       <value xsi:type="am:LongObject" value="2" />
        </customProperties>
464   </literals>
      <literals name="Message_3">
466     <customProperties key="enumValue">
          <value xsi:type="am:LongObject" value="3" />
468     </customProperties>
      </literals>
470   <literals name="Message_4">
        <customProperties key="enumValue">
472       <value xsi:type="am:LongObject" value="4" />
        </customProperties>
474   </literals>
    </modes>
```

```
476     <modes name="MessageToT1">
          <literals name="MessageToT1_0">
478         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
480         </customProperties>
          </literals>
482       <literals name="MessageToT1_1">
            <customProperties key="enumValue">
484           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
486       </literals>
        </modes>
488     <modes name="MessageToT2">
          <literals name="MessageToT2_0">
490         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
492         </customProperties>
          </literals>
494       <literals name="MessageToT2_1">
            <customProperties key="enumValue">
496           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
498       </literals>
        </modes>
500     <modes name="StateT1">
          <literals name="StateT1_0">
502         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
504         </customProperties>
          </literals>
506       <literals name="StateT1_1">
            <customProperties key="enumValue">
508           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
510       </literals>
        </modes>
512     <modes name="StateT2">
          <literals name="StateT2_0">
514         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="0" />
516         </customProperties>
          </literals>
518       <literals name="StateT2_1">
            <customProperties key="enumValue">
520           <value xsi:type="am:LongObject" value="1" />
            </customProperties>
522       </literals>
          <literals name="StateT2_2">
524         <customProperties key="enumValue">
              <value xsi:type="am:LongObject" value="2" />
526         </customProperties>
          </literals>
528     </modes>
        <modeLabels name="message" initialValue="Message/Message_0?type=ModeLiteral">
530       <size value="8" unit="bit" />
        </modeLabels>
532     <modeLabels name="messageToT1" initialValue="MessageToT1/MessageToT1_0?type=ModeLiteral">
          <size value="1" unit="bit" />
```

```
534     </modeLabels>
        <modeLabels name="messageToT2" initialValue="MessageToT2/MessageToT2_0?type=ModeLiteral">
536       <size value="1" unit="bit" />
        </modeLabels>
538     <modeLabels name="stateT1" initialValue="StateT1/StateT1_1?type=ModeLiteral">
          <size value="1" unit="bit" />
540     </modeLabels>
        <modeLabels name="stateT2" initialValue="StateT2/StateT2_0?type=ModeLiteral">
542       <size value="8" unit="bit" />
        </modeLabels>
544   </swModel>
      <hwModel>
546     <definitions xsi:type="am:ProcessingUnitDefinition" name="DefaultCore" features="Instructions/
            IPC_1.0?type=HwFeature" puType="CPU"/>
        <definitions xsi:type="am:MemoryDefinition" name="DefaultMemory">
548     </definitions>
        <featureCategories name="Instructions" featureType="performance">
550       <features name="IPC_1.0" value="1.0" />
        </featureCategories>
552     <structures name="System" structureType="System">
          <structures name="Ecu_1" structureType="ECU">
554         <structures name="Processor_1" structureType="Microcontroller">
              <modules xsi:type="am:Memory" name="Memory_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultMemory?type=MemoryDefinition">
556           </modules>
              <modules xsi:type="am:ProcessingUnit" name="Core_1" frequencyDomain="Frequency_1?type=
                  FrequencyDomain" definition="DefaultCore?type=ProcessingUnitDefinition">
558             <ports name="port" bitWidth="32" priority="0" portType="initiator"/>
              </modules>
560         </structures>
          </structures>
562     </structures>
        <domains xsi:type="am:FrequencyDomain" name="Frequency_1" clockGating="false">
564       <defaultValue value="600.0" unit="MHz"/>
        </domains>
566   </hwModel>
      <osModel>
568     <operatingSystems name="Generic_OS">
          <taskSchedulers name="Scheduler_1">
570         <schedulingAlgorithm xsi:type="am:OSEK" />
          </taskSchedulers>
572       <osDataConsistency mode="noProtection" />
        </operatingSystems>
574   </osModel>
      <stimuliModel>
576     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_1">
          <offset value="0" unit="ms" />
578       <recurrence value="220" unit="ms" />
        </stimuli>
580     <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_2" />
        <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_3">
582       <offset value="0" unit="ms" />
          <recurrence value="50" unit="ms" />
584     </stimuli>
        <stimuli xsi:type="am:InterProcessStimulus" name="IPA_Task_4" />
586     <stimuli xsi:type="am:PeriodicStimulus" name="Stimulus_Task_5">
          <offset value="0" unit="ms" />
588       <recurrence value="500" unit="ms" />
```

```
    </stimuli>
  </stimuliModel>
  <constraintsModel />
  <eventModel>
    <events xsi:type="am:ProcessEvent" name="Event_Task_1" entity="Task_1?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_2" entity="Task_2?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_3" entity="Task_3?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_4" entity="Task_4?type=Task" />
    <events xsi:type="am:ProcessEvent" name="Event_Task_5" entity="Task_5?type=Task" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1" entity="Runnable_1?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_0" entity="Runnable_1_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_1" entity="Runnable_1_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State0" entity="Runnable_1_State0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_1_State1" entity="Runnable_1_State1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_2_Overflow" entity="
        Runnable_2_Overflow?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3" entity="Runnable_3?type=Runnable"
        />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_0" entity="Runnable_3_0?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_1" entity="Runnable_3_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_2" entity="Runnable_3_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_3" entity="Runnable_3_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_3_4" entity="Runnable_3_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_1" entity="Runnable_4_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_2" entity="Runnable_4_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_3" entity="Runnable_4_3?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_4" entity="Runnable_4_4?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_4_x" entity="Runnable_4_x?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_1" entity="Runnable_5_1?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_5_2" entity="Runnable_5_2?type=
        Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_0" entity="Runnable_State_0?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_1" entity="Runnable_State_1?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_State_2" entity="Runnable_State_2?
        type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_0" entity="
        Runnable_Transition_0?type=Runnable" />
    <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_1" entity="
        Runnable_Transition_1?type=Runnable" />
```

```
622     <events xsi:type="am:RunnableEvent" name="Event_Runnable_Transition_2" entity="
            Runnable_Transition_2?type=Runnable" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_1" entity="Stimulus_Task_1?type=
            PeriodicStimulus" />
624     <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_2" entity="IPA_Task_2?type=
            InterProcessStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_3" entity="Stimulus_Task_3?type=
            PeriodicStimulus" />
626     <events xsi:type="am:StimulusEvent" name="Event_IPA_Task_4" entity="IPA_Task_4?type=
            InterProcessStimulus" />
        <events xsi:type="am:StimulusEvent" name="Event_Stimulus_Task_5" entity="Stimulus_Task_5?type=
            PeriodicStimulus" />
628   </eventModel>
      <mappingModel addressMappingType="offset">
630     <taskAllocation task="Task_1?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_2?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
632     <taskAllocation task="Task_3?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <taskAllocation task="Task_4?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
634     <taskAllocation task="Task_5?type=Task" scheduler="Scheduler_1?type=TaskScheduler" />
        <schedulerAllocation scheduler="Scheduler_1?type=TaskScheduler" responsibility="Core_1?type=
            ProcessingUnit" />
636     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="11" abstractElement="
            stateT2?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="10" abstractElement="
            stateT1?type=ModeLabel" />
638     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="8" abstractElement="
            messageToT1?type=ModeLabel" />
        <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="0" abstractElement="
            message?type=ModeLabel" />
640     <memoryMapping memory="Memory_1?type=Memory" memoryPositionAddress="9" abstractElement="
            messageToT2?type=ModeLabel" />
      </mappingModel>
642   <componentsModel />
    </am:Amalthea>
```

Listing A.36: Variation 7 of State Machine Feedback Loop.

University
of Bamberg
Press

Model-driven approaches are experiencing an increasing acceptance in the automotive domain thanks to the availability of the AUTOSAR standard. However, the process of creating models of existing system components is often difficult and time consuming, especially when legacy code is involved or information about the exact timing is needed.

This work focuses on reversely engineering an AUTOSAR-compliant model, which can be used for further processing including timing simulation and optimisation, via a dynamic analysis from trace recordings of a real-time system. Huselius, whose work is among the publications most related to the topic of this thesis, proposes a technique to reverse engineer a model that reflects the general temporal behaviour of the original real-time software. However, like other existing solutions, it was not developed with AUTOSAR in mind.

We want to tackle this deficiency by introducing an approach that seizes on Huselius's considerations and extends them in order to make them applicable to the automotive domain. To do so, we present CoreTAna, a prototypical tool that derives an AUTOSAR compliant model of a real-time system by conducting dynamic analysis using trace recordings. Motivated by the challenge of assessing the quality of reverse engineered models of real-time software, we also introduce a mathematical measure for comparing trace recordings from embedded real-time systems regarding their temporal behaviour and a benchmark framework based on this measure, for evaluating reverse engineering tools such as CoreTAna.

eISBN: 978-3-86309-691-5

9 783863 096915

www.uni-bamberg.de/ubp